PCT

WORLD INTELLECTUAL PROPERTY ORGANIZATION
International Bureau

INTERNATIONAL APPLICATION PUBLISHED UNDER THE PATENT COOPERATION TREATY (PCT)

(51) International Patent Classification 7 :
H04N 7/12, 7/173, 9/74, G11B 31/00, G06F 17/00

A1

(11) International Publication Number: WO 00/64168

(43) International Publication Date: 26 October 2000 (26.10.00)

(21) International Application Number: PCT/US00/10499

(22) International Filing Date: 19 April 2000 (19.04.00)

(30) Priority Data:
60/130,059 19 April 1999 (19.04.99) US

(71) Applicant: I PYXIDIS LLC [US/US]; Suite 205, 7441 Sunset Boulevard, Los Angeles, CA 90046 (US).

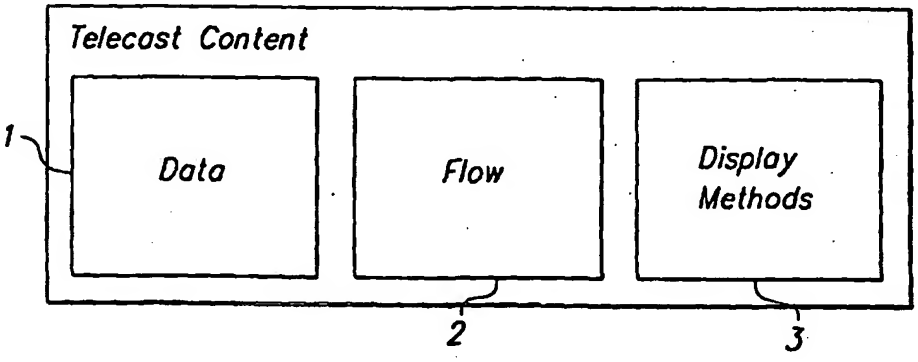
(72) Inventors: KAUFMAN, Christopher, D.; 1811 Courtney Avenue, Los Angeles, CA 90046 (US). BELCHER, Chris; 3329 Keystone Avenue #5, Los Angeles, CA 90034 (US). KOSKY, Curtis, R.; 514 S. Barrington Avenue #17, Los Angeles, CA 90049 (US).

(74) Agents: BEAMER, Norman, H. et al.; Fish & Neave, 1251 Avenue of the Americas, New York, NY 10020 (US).

(81) Designated States: AE, AG, AL, AM, AT, AU, AZ, BA, BB, BG, BR, BY, CA, CH, CN, CR, CU, CZ, DE, DK, DM, DZ, EE, ES, FI, GB, GD, GE, GH, GM, HR, HU, ID, IL, IN, IS, JP, KE, KG, KP, KR, KZ, LC, LK, LR, LS, LT, LU, LV, MA, MD, MG, MK, MN, MW, MX, NO, NZ, PL, PT, RO, RU, SD, SE, SG, SI, SK, SL, TJ, TM, TR, TT, TZ, UA, UG, UZ, VN, YU, ZA, ZW, ARIPO patent (GH, GM, KE, LS, MW, SD, SL, SZ, TZ, UG, ZW), Eurasian patent (AM, AZ, BY, KG, KZ, MD, RU, TJ, TM), European patent (AT, BE, CH, CY, DE, DK, ES, FI, FR, GB, GR, IE, IT, LU, MC, NL, PT, SE), OAPI patent (BF, BJ, CF, CG, CI, CM, GA, GN, GW, ML, MR, NE, SN, TD, TG).

Published
With international search report.

(54) Title: METHODS AND APPARATUS FOR DELIVERING AND VIEWING DISTRIBUTED ENTERTAINMENT BROADCAST OBJECTS AS A PERSONALIZED INTERACTIVE TELECAST



(57) Abstract

The present invention relates to methods and apparatus for delivering, customizing, and displaying personalized media content, (Fig. 4, Fig. 10). Content is divided into data, flow, and display aspects (Fig 1), each of which may be manipulated to alter a television–style show in real–time. Viewer system apparatus includes set top box, advanced TV, personal computer, or media appliance. Interactive shows (Fig. 11) may be implemented by altering the data, flow or display of the content in real–time at a server, according to communications from viewers (Fig. 10) or according to business needs.

BEST AVAILABLE COPY

FOR THE PURPOSES OF INFORMATION ONLY

Codes used to identify States party to the PCT on the front pages of pamphlets publishing international applications under the PCT.

AL	Albania	**ES**	Spain	**LS**	Lesotho	**SI**	Slovenia
AM	Armenia	**FI**	Finland	**LT**	Lithuania	**SK**	Slovakia
AT	Austria	**FR**	France	**LU**	Luxembourg	**SN**	Senegal
AU	Australia	**GA**	Gabon	**LV**	Latvia	**SZ**	Swaziland
AZ	Azerbaijan	**GB**	United Kingdom	**MC**	Monaco	**TD**	Chad
BA	Bosnia and Herzegovina	**GE**	Georgia	**MD**	Republic of Moldova	**TG**	Togo
BB	Barbados	**GH**	Ghana	**MG**	Madagascar	**TJ**	Tajikistan
BE	Belgium	**GN**	Guinea	**MK**	The former Yugoslav Republic of Macedonia	**TM**	Turkmenistan
BF	Burkina Faso	**GR**	Greece			**TR**	Turkey
BG	Bulgaria	**HU**	Hungary	**ML**	Mali	**TT**	Trinidad and Tobago
BJ	Benin	**IE**	Ireland	**MN**	Mongolia	**UA**	Ukraine
BR	Brazil	**IL**	Israel	**MR**	Mauritania	**UG**	Uganda
BY	Belarus	**IS**	Iceland	**MW**	Malawi	**US**	United States of America
CA	Canada	**IT**	Italy	**MX**	Mexico	**UZ**	Uzbekistan
CF	Central African Republic	**JP**	Japan	**NE**	Niger	**VN**	Viet Nam
CG	Congo	**KE**	Kenya	**NL**	Netherlands	**YU**	Yugoslavia
CH	Switzerland	**KG**	Kyrgyzstan	**NO**	Norway	**ZW**	Zimbabwe
CI	Côte d'Ivoire	**KP**	Democratic People's Republic of Korea	**NZ**	New Zealand		
CM	Cameroon			**PL**	Poland		
CN	China	**KR**	Republic of Korea	**PT**	Portugal		
CU	Cuba	**KZ**	Kazakstan	**RO**	Romania		
CZ	Czech Republic	**LC**	Saint Lucia	**RU**	Russian Federation		
DE	Germany	**LI**	Liechtenstein	**SD**	Sudan		
DK	Denmark	**LK**	Sri Lanka	**SE**	Sweden		
EE	Estonia	**LR**	Liberia	**SG**	Singapore		

METHODS AND APPARATUS FOR DELIVERING AND VIEWING DISTRIBUTED ENTERTAINMENT BROADCAST OBJECTS AS A PERSONALIZED INTERACTIVE TELECAST

Field Of The Invention

The present invention relates to methods and apparatus for delivering, customizing, and displaying a personalized interactive telecast over a broadcast medium, or over the Internet.

Background Of The Invention

Over the last two decades, there have been numerous attempts to deliver interactive or customizable television-style content. The first of these attempts were a variety of "video on demand" systems, that would allow a user to select a movie, and view it immediately on their television. Though such systems seemed conceptually feasible, there was never any widespread deployment or use of such systems, due to the relatively poor video compression technology available at the time, and the prohibitive expense of assembling large scale video servers.

The initial work on "video on demand" led to the development of "pay-per-view" movies, which permit a cable company or other video supplier to charge viewers to play a particular movie. Pay-per-view systems, however, are not interactive or customizable,

since the user is unable to control the flow of a movie, e.g. by pausing, fast-forwarding, or rewinding the movie. Instead, pay-per-view movies generally run continuously once started, just like any other television or cable channel, giving the viewer no control over the manner in which the movie is played.

With the advent of the Internet, attempts were made to combine Internet technologies with a standard analog television signal, such as an NTSC, PAL, or SECAM signal. Data was sent using a section of the analog television signal known as the "vertical blanking interval," or VBI. The VBI is a portion of the signal that is not normally visible on the television screen, and has been used for many years to carry digital data, such as the text for closed captioning. An NTSC signal, for example, contains 525 scanlines, the last 25 of which are the VBI. Line 21 of the VBI in an NTSC signal is used to store closed captioning text, which may be displayed to assist hearing-impaired viewers.

Numerous companies, including Microsoft's WebTV and Intel, currently offer products that are able to extract data from the VBI, and Intel and Microsoft have started a standards body known as the Advanced TV Enhancement Forum (ATVEF) to develop standards for the delivery of data in a television signal. At present, data is inserted into the television signal using a "VBI insertion device," which inserts Internet Protocol (IP) multi-cast packets into the VBI. Multi-cast packets are a "one way" delivery mechanism, in that there is no way to insure against dropped or lost packets of data.

Software applications have been developed to use the multi-cast packets in the VBI signal to display

web pages, similar to the web pages that may be displayed by a conventional web browser, such as Netscape Navigator, or Microsoft's Internet Explorer. Additionally, this software permits some limited synchronization of the web pages with the video. Basically, video is embedded in a web page with text headlines. The display of such web pages can be scheduled by the broadcaster, so that a particular web page is sent to the VBI inserter at a pre-determined time. Although such synchronization is possible, the text and video in these systems are separate. The text or web-page portion has no "knowledge" of the content of the video, and the video is "unaware" that it is being displayed in the context of a web page. Additionally, such previously known systems provide no ability for a user to interact with the video or textual content, or to customize the content.

Companies have had only limited success with products that add web pages to television. With little or no interaction, and no customization features, the products based on this technology have not been sufficiently compelling to attract many customers. Additionally, because television sets have lower resolution and quality than computer monitors, web content presented on television screens is difficult to read. Further, the Document Object Model (DOM) employed by web pages is not well suited to delivering interactive television, since it is based on the concept of pages and documents, rather than on sequences, as is most television content.

Other attempts to deliver interactive television-like entertainment are entirely Internet-based. A user receives a digital video stream over the Internet, and displays the video. Additionally, the

digital video stream may contain various scripted command "events" that are synchronized to the video, and that cause messages or other multimedia content to be displayed. Since these systems require an Internet connection, which permits bidirectional communication, a greater level of interaction and customization are possible than with systems that have only a "one way" link through an analog video signal. Both Microsoft Corporation, of Redmond, Washington, and RealNetworks, Inc., of Seattle, Washington, provide streaming video products that access digital video streams over the Internet, and that permit limited synchronization of events with the video stream.

Unfortunately, most homes do not currently have Internet connections that are fast enough to permit television-quality full-screen video to be sent over the Internet. Current dial-up connection speeds provide a bandwidth of only 28.8 to 56 kilobits per second, which is far less than the approximately 1 to 1.5 megabits per second sustained data rate required to send full screen television-quality video and stereo sound over the Internet. Even more recent "high-speed" Internet connection technologies, such as DSL and cable modems only reach speeds of 500 kilobits to 1.5 megabits per second, and have difficulty sustaining a connection above the required level of approximately 1 megabit per second. Given this lack of bandwidth to the home, for most television viewers, a purely Internet-based interactive television is simply not feasible.

Additionally, the type of interactivity and customization offered by currently-available Internet-based streaming video products is limited. Although these systems support "command events" synchronized

with the video, these events are "hard coded," or built into the video at design time. Thus, prior to releasing an interactive video segment, the events must be programmed to synchronize with the video using an unchangeable, and non-dynamic script form. Once such a script is created, the viewer or broadcaster has no ability to change the hard coded events of the video, to change the sequence of events, or to alter or manipulate the digital video. Although some of these shortcomings may be addressed by using known dynamic scripting techniques, such dynamic scripting is typically unstable, slow, cumbersome, and highly platform-dependent.

Other recent devices, such as digital VCRs, offer some limited interactive capabilities. Digital VCRs permit a user to save video digitally, and permit viewing of the video with the ability to rewind, fast forward, pause, etc. Video is saved in "folders" similar to the folder system of a Windows PC. The user can create custom folders and play back the video from those folders. The video contained in these folders is not altered, updated, or customized other than being saved in a static digital format.

PCs with a TV tuner card may have the same capabilities of these Digital VCRs. For example, Intel has included a set of special video encoding and decoding instructions in their Pentium III processors. These built in instructions enable PCs with a Pentium III processor to encode and decode MPEG2 video using the processor rather than an extra layer of software. Most PCs sold in the next five years will have Digital VCR features built into the system.

In view of the above, it would be desirable to provide methods and apparatus for delivering and viewing personalized, interactive telecast entertainment.

It would further be desirable to provide methods and apparatus that permit personalized, interactive telecast entertainment to be delivered using either a broadcast or cable medium, or an Internet connection.

It would also be desirable to provide methods and apparatus that enable a user, business, or broadcaster to dynamically alter the sequence, flow, or content of a telecast, either in advance, or as it is being broadcast.

Summary Of The Invention

It is an object of the present invention to provide methods and apparatus for delivering and viewing interactive and customizable telecast entertainment.

It is a further object of this invention to provide methods and apparatus that permit interactive and customizable telecast entertainment to be delivered using either a broadcast or cable medium, or an Internet connection.

It is another object of the present invention to provide methods and apparatus that enable a user, business, or broadcaster to dynamically alter the sequence, flow, or content of a telecast, either in advance, or as it is being broadcast.

These and other objects of the present invention are achieved by providing telecast content that is separated into reusable accessible object containers, referred to as (1) data, which contains the

various components and objects needed to assemble the content, (2) flow information, which describes the flow or sequence of the content, and (3) display methods, that provide information on how the content is to be displayed. These reusable accessible object containers are referred to as "Distributed Entertainment Broadcast Objects" (DEBO), and are represented using a hierarchical markup language called WSML. By separating the content into these DEBOs, a high degree of customization and personalization may be achieved simply by altering or inserting items into the data, the flow, or the display methods. These alterations may be made from a server or on a client (i.e. a set-top box, personal computer, or other media appliance) in a viewer's home, using a one-way broadcast data stream which may be inserted into the vertical blanking interval of a television signal, sent in a digital television signal, or sent over the Internet.

Additionally, if a two-way communications network, such as the Internet, is used, the present invention provides for a high degree of interactivity through use of a centralized server, that may change the broadcast data, flow, or display according to information received from viewers. This ability to change the broadcast in response to communications from the viewers may be used to implement interactive programing, such as an auction show, a chat show, or interactive game shows.

Brief Description Of The Drawings

The above and other objects and advantages of the present invention will be apparent upon consideration of the following detailed description, taken in conjunction with the accompanying drawings, in

which like reference characters refer to like parts throughout, and in which:

FIG. 1 shows telecast content split into data, flow, and display methods;

FIGS. 2A-C show samples of data, flow, and display methods, respectively;

FIG. 3 shows the structure of server-side transaction engines in accordance with the present invention;

FIG. 4 shows the structure of a client built in accordance with the principles of the present invention;

FIG. 5 shows the contents of a webshow index file;

FIG. 6 shows the structure of a server built in accordance with the principles of the present invention;

FIG. 7 shows the interconnection of the server and client;

FIG. 8 shows the interaction of the client with the store and distribute transaction services of the server;

FIG. 9 illustrates types of transactions between the server and client;

FIG. 10 shows the structure of the webshow user interface; and

FIG. 11 shows the structure of the datatube control.

Detailed Description Of The Invention

The present invention provides methods and apparatus for delivering, customizing, and displaying entertainment based telecasts using a system comprising a client application and a server application. The

client application handles personalization and display of content; the server application handles delivery of content, and interactions and transactions with viewers. The system provides for customization at both the client side and the server side, and delivers "personal telecasts" to viewers. It should be noted that as used herein, a telecast includes any means by which media content may be provided to a client, including broadcast, Internet, cable, CD-ROM, and DVD.

In the context of the present invention, a "personal telecast" is a combination of standard telecast content (either analog or digital), with a specialized content structure that contains information on the flow, data, and display methods for a show. This specialized content structure, called a "parsed datastream", is a continuous broadcast data feed from a server to a client, that is broken down into accessible, reusable objects to permit customization of telecast entertainment at both the server and client. Additionally, a parsed datastream may provide security features, such as encryption. These security features enable broadcasters to provide pay-per-view programming, blocking, and to limit access to specific content or content depth.

The information contained in a parsed datastream may be sent in the vertical blanking interval of a standard television broadcast, in a digital television signal, or may be sent over some other communication medium, such as the Internet. Information on the viewer, such as the viewer's name and demographic information, may be present on the client side (i.e. in a set-top box, personal computer, specialized television set, etc.), and may be used to filter material from the broadcast data to personalize

content. If multiple viewers typically use a single client device (i.e. a set-top box or personal computer), a "login" screen may be used to determine which viewer is the primary viewer at any given time, so that content may be customized to that viewer's tastes.

As shown in FIG. 1, the present invention represents television-style content as a collection of objects that can be parsed into three portions -- data 1, flow 2, and display methods 3. This separation of the content permits a high degree of customization of television-style content to be achieved in a broadcast environment, in which a "back channel" permitting two-way communication may not be available. Various alterations to the data, flow, or display may be made at the client end to permit a viewer to customize the content displayed on his or her television, personal computer, or other media appliance. Alterations to data or flow made on the server may affect all viewers, or some subset of viewers, providing television stations or networks with the ability to easily change broadcast content, or to alter broadcasts according to feedback from viewers, or according to demographic information on the viewers.

Data 1 includes all of the "raw" data of a television show, or other telecast content, separated into various components or objects. These objects may include digital video, digital audio, animation, three-dimensional characters, music, still images, text, and other types of digital data. For example, the components for a news broadcast may include images of one or more announcers or anchors, the dialog associated with each of the announcers (in both text and sound files), background or introduction music,

live video images, and a "placeholder" for other media elements.

In a preferred embodiment, there may be alternatives available for one or more of these components. For example, the data for a news program may include imagery and sound for both a male anchor and a female anchor, reading the same news stories. A user's preferences may determine which of the anchors is displayed during the news program. Additionally, a preferred embodiment permits data 1 to be altered according to a viewer's preferences or input to achieve a degree of customization. For example, by adding a viewer's name to the dialog data in a show, the characters in the show may address or greet the viewer by name.

Alteration of data 1 on the server side may be used by stations, networks, or businesses to change the information that is displayed across all of the viewers of a show. As will be described in greater detail hereinbelow, this ability could be used to provide interactive features, such as auctions or chats, in which data collected from multiple viewers must be transmitted to all of the viewers of the auction or chat program.

Data 1 is preferably encoded in a hierarchical format, such as XML, as will be discussed in greater detail hereinbelow. Such a hierarchical or tree format permits random access to the data, and permits the data to be easily classified, stored, and altered. By encoding data 1 in such a hierarchical format, it is possible to easily change individual objects without employing complex scripting techniques.

Flow 2 describes the flow, or sequence of a television show or other telecast content. Basically,

flow 2 comprises a list of actions to be taken sequentially, and the data on which the actions are to be taken. The flow for a news program, for example, might specify that an image for an anchorman is displayed, introduction dialog is played, dialog for a first news story is played, video for the first news story is played, dialog for a second news story is played, and so on. Flow 2 preferably contains "bookmarks," that permit specific portions of a program to be named and tagged for easy access. The flow for a news program, for example, might include "bookmarks" identifying a national news section, an international news section, a business section, a sports section, a weather section, and so on.

By altering flow 2 on the client, television shows or other content may be personalized to meet the preferences of a particular viewer. Sections of a show may readily be skipped or rearranged to personalize the show. For example, if a viewer of a news program is primarily interested in sports, the flow of the news program could be altered at the client side to place sports first in the news program. If the viewer is only interested in sports, the flow could be altered to skip everything in the news broadcast except the sports section. Additionally, stepping forward or backward through the flow may be used to create a fast forward or rewind effect, and temporarily stopping the flow may be used to pause a program.

On the server side, flow 2 may be altered by stations or networks to rearrange their schedules, to insert special bulletins or announcements, or to alter the schedule according to feedback from viewers or viewer demographics. Additionally, alterations to flow 2 may be made on the server to assist in providing

interactive programs, such as an auction, a chat show, or a mystery show, in which the ending is selected by taking a vote of the viewers.

Display methods 3 specifies the platform-dependent methods and plugins that should be used to display each type of content that makes up a television show or other telecast content. Thus, display methods 3 may comprise a file that specifies for each type of client on which shows may be displayed (e.g. television set-top boxes, personal computers, etc.), which plug-in will display each type of content. For example, display methods 3 may specify that video content in a show displayed on a set-top box should be displayed using the "video-player" plugin.

By specifying display methods 3 in this manner, the methods and apparatus of the present invention are able to handle displaying the same content on a wide variety of platforms. For example, the same telecast content may be viewed on a television set, using a set-top box receiving the broadcast over cable, or on a personal computer, receiving the telecast over the Internet. This is achieved by specifying the appropriate plugin to be used on each of these devices for each of the types of content present in a show. The display of the content may vary, according to the capabilities of the display device.

Similarly, the ability to specify display methods 3 permits the system of the present invention to be upgraded or expanded to handle new types of content. The viewer would need to download (possibly through a special "upgrade" broadcast) a plugin that handles the new content type. The system could then include a method specifying the new plugin as the correct display method for the new type of content.

Thus, as new content types, such as 3-D video and sound, become more widely available, the new plugins can be used to display them. Additionally, changing display methods 3 permits a viewer to personalize the manner in which telecast content is displayed.

Referring now to FIG. 2A, a portion of data 1 for a sports program is shown. In this example, XML tags are used to mark the various components of data 1. The components of this particular example include announcers, dialog, images for the announcers, and music. The use of XML tags permits the data to be organized according to its type, and to be randomly accessed. Although XML is used in the preferred embodiment, any format that permits data to be organized by type and tagged for retrieval could be used.

FIG. 2B shows an example of flow 2 for a sports show using the data example from FIG. 2A. XML is used to specify the order of the events. As will be understood by one skilled in the relevant arts, other formats may also be used to specify the order in which actions are to be taken. In this example, the system loads the first announcer ("Announcer1"), displays an image of the first announcer, and directs the first announcer to speak the first portion of the intro (using, e.g., text-to-speech software). Next, flow 2 of FIG. 2B directs the system to load the second announcer, display the second announcer, and speak the second portion of the intro.

FIG. 2C shows an example of display method 3. XML is used here to specify the video player that is to be used on various platforms. As will be understood, other data formats could also be used to specify the information of display 3. In the example shown in FIG.

2C, if the show is played on a Macintosh computer, the "QuickTime" plugin will be used to play video. If the show is played on a PC, the "NetShow" plugin will be used, and so on.

In a preferred embodiment, the XML tags described above that are used to mark the various components of data, flow, and display are based upon Web Show Markup Language, or WSML. WSML, like other markup languages such as HTML, and Synchronized Markup Internet Language (SMIL), all derive from Standard Generalized Markup Language, or SGML. More particularly, both SMIL and WSML derive from XML, a sub-language of SGML.

However, WSML differs from other markup languages in that it uses time as a subcomponent. WSML is based on events, methods, functions, and properties. HTML, the most well-known markup language, does not use time as a component. SMIL is similar to WSML in that it uses time as a component. However, SMIL is only able to describe a single linear timetable of events. WSML, on the other hand, uses time as a subcomponent, and can create branches and add timetables of events. WSML can support multiple branching timelines, whereas SMIL is not structured to do so.

As described above, the separation of telecast content into data, flow, and display objects (i.e. DEBOs) permits customization, personalization, and a small degree of interactivity to be achieved on a system that does not have a "back channel" permitting two-way communication. Where such two-way communication is possible, using either an Internet connection, or other means for sending information from a viewer's location to a server that delivers the show content, a greater degree of interactivity, including

interaction between viewers, may be achieved. Further, by separating telecast content into data, flow, and display objects (DEBOs), a significantly greater degree of reusability for the individual objects is achieved, because objects are no longer tied together through techniques that would hardcode all logic or dynamic elements into a single document, such as is done in HTML, DHTML, Cold Fusion Markup Language, or other languages based on a document object model. Similarly, separation of telecast content into data, flow, and display provides a greater degree of flexibility than is achieved using traditional programming techniques, which typically employ hard-coded logic and interactions. Finally, such separation of data, flow, and display methods into distributed entertainment broadcast objects (DEBOs) allows for changes to be made in real-time, as the user is viewing the telecast content. For example, by separating the flow into a series of accessible event lists, a user can add or subtract to these event lists as the telecast content is being broadcast. By separating the display methods, as opposed to tying them to a single document, a user can now determine what media is sent to a particular player without having to reload or refresh objects, as opposed to simply playing the media hard coded into a web page. By separating the data into a set of objects with properties, a user can seamlessly replace that data with other data having the same type of properties. It should also be noted that most any changes that may be made by a user may also be made at the server by a broadcaster or business, to customize the content to meet the needs of the broadcaster or business, providing broadcasters and businesses with a way to provide customizable interactive content.

Referring to FIG. 3, further in accordance with the present invention, a system that permits interactivity using three levels of "transaction engines" running on a server is described. A station transaction engine 4 is associated with a plurality of show transaction engines 5, each of which is optionally associated with a plurality of room transaction engines 6.

The lowest level transaction engines are room transaction engines 6. Each room transaction engine 6 handles transactions for a particular show associated with viewers having a particular demographic characteristic, such as a particular postal code, gender, age, income level, etc. When viewers from within the specific demographic send information to the system, that information is filtered by the room transaction engine before being forwarded to a higher level (i.e. show) transaction engine. For example, for a show that determines the ending according to votes, there may be logic at the room transaction engine level that only forwards votes from the demographic represented by the room transaction engine if there are more than 1000 votes. For shows such as an auction, the room transaction engine may send all bids received for an item on to the show transaction engine, or may send only information on the high bid. Similarly, the ability to process data at the "room" level may be used to implement off-site learning programs.

The logic involved in making these decisions on routing of information from the room transaction engine to higher level transaction engines is preferably hard coded, and may be dependent on the show, and the targeted audience. Code that implements the needed logic may be loaded as a server behavior

plugin, such as a TV logic plugin as described hereinbelow, for the room transaction engine.

Room transaction server engine 6 includes a sequencer that may be used to customize the data, flow, or display according to the demographic assigned to a particular room. This can be used to insert local teasers, giveaways, contests, or ads into the flow, as well as providing a method to send local disclaimers, or messages meant to conform to local law. For example, in a "room" associated with a particular postal code, advertisements for a car dealer local to that postal code could be inserted into the flow. Similarly, if the room demographic is children, the flow for certain portions of programs may be altered to make the programming more educational, and suitable for children, and ads for toys or pre-sweetened cereal could be inserted into the flow, rather than ads for cars.

The next level of transaction engines are show transaction engines 5. Show transaction engines 5 handle transactions associated with particular shows, such as auction shows, or chat shows. Like room transaction engines 6, show transaction engines 5 may use hard-coded logic contained in a server behavior plugin to filter information from viewers, and determine how that information is to be handled.

The logic in a show transaction engine may cause the data, flow, or display of an entire show to be altered according to transactions with viewers. For example, the show transaction engine for a show in which viewers vote for a desired ending would count up the votes (received either directly, or from one or more room transaction servers), and cause a show-level sequencer on the server to insert the "winning" ending

into the show. Similarly, for an auction show, the show transaction engine would determine the high bid at any given time, and determine, based on the bidding, when the auction ends, whether to give more detail on the item up for auction, whether to go on to another item, and so on. Information on the current highest bid would be inserted into the data and flow of the auction show. When the auction for an item ends, the auction show transaction engine would determine who "won" the item in the auction, and would cause information on the final price and "winner" to be inserted into the auction show's data and flow. Chat shows, and multi-player game shows also may be implemented in a similar manner.

Show transaction engine 5 includes a sequencer that may be used to customize the data, flow, or display of a show. This can be used to insert show-specific teasers, giveaways, contests, quizzes, or ads into the flow, as well as providing a method to send disclaimers or other legal messages associated with a particular show.

The highest level transaction engine is station transaction engine 4. Station transaction engine 4 handles station-wide transactions, and may be implemented using hard-coded logic in a server behavior plugin. Station transaction engine 4 includes a sequencer, and handles station-wide transactions and scheduling, including teasers, giveaways, station identification breaks, emergency breaks, and station disclaimers and other legal messages associated with the station.

Referring now to FIG. 4, the structure of a client system for providing personalization and display of telecast content is described. Client 7 may

comprise a set-top box, a personal computer, a television set containing specialized electronics, a media appliance, or any other home device that may receive, process, and display television-style content. Client 7 receives telecast content via broadcasts over the air, through cable, over the Internet, or via any other medium.

The following is a description of how a preferred embodiment of client 7 displays and personalizes telecast content. A software component of client 7, called webshow engine 8, is exposed for external use. When the user wishes to view telecast content on client 7, the user activates webshow engine 8 by way of scripting or other similar methods. Once activated, webshow engine 8 in turn instantiates its three sub-components: data control 9, index manager 10, player control 11.

Data control 9 handles all of data 1 needed by client 7. Data control 9 receives the data portion of a parsed datastream, as described hereinabove with reference to FIGS. 1 and 2A, and stores data 1 on a storage device in client 7. Data Control 9 also stores information about the viewer's preferences and personalization data. Additionally, data control 9 may store digitized television signals on client 7, so that they may be retrieved and manipulated after they are broadcast.

Data control 9 includes data resource manager 12. Data resource manager 12 is responsible for validating the integrity of the data, and insuring that all of the needed data for display is present when it is needed. Since data control 9 contains objects which can affect flow 2, it can anticipate which data will be needed, or make substitutions in the data if necessary.

Thus, by replacing these objects, the flow may be altered as necessary.

For example, if data resource manager 12 determines that some portion of the needed data is not present, or has been corrupted or missed during the broadcast process, it may be able to connect to the Internet to retrieve the needed data. Alternatively, if the data will be repeated at some later time in the broadcast, it may be possible to retrieve the missed data at that time. Thus, data resource manager 12 provides a degree of data integrity to broadcast content.

If the needed data cannot be retrieved, data resource manager 12 may alter the flow to substitute in data that is similar in nature to the missing data. For example, if a user attempts to fast-forward in a sports broadcast that includes live video data of a basketball game in progress, data resource manager 12 will be unable to retrieve the required video data (since the data does not yet exist). Depending on the viewer's preferences, data resource manager 12 may alter the flow to show a screen explaining the unavailability of the data, a screen showing a still image from earlier in the basketball game, or video from earlier in the game, or from previous games.

Index manager 10 handles flow 2 of programs on client 7, based on flow scripts, as described hereinabove with reference to FIGS. 1 and 2B. Based on a viewer's commands or preferences, portions of the flow may be altered, skipped, or rearranged. By allowing access to portions of the flow only if the user has paid, stations may implement pay-per-view programming. Additionally, client-side parental controls and preferences may be used to for blocking,

by causing index manager 10 to remove or skip portions of the flow that are marked as being unsuitable for children. Index manager 10 displays content using player control 11, and gets the data it requires from data control 9.

Player control 11 deals with the actual displaying of the interactive telecast content, called a "webshow", by handling the various display and interaction plugins available on client 7. For example, when video must be displayed, player control 11 will load a plugin that is capable of displaying video, as specified in the display portion of a parsed datastream (described hereinabove with reference to FIGS. 1 and 2C), and send the video to the plugin for display.

In a preferred embodiment, the set of plugins handled by player control 11 includes display plugins, that may be used to overlay and layer media content, including digital video. This display plugin takes digital video, and turns a range of colors in the digital video transparent, which permits multiple layers of digital video to be assembled, with the transparent areas of each layer showing the contents of the layer behind it. The set of plugins handled by player control 11 also preferably include plugins for handling video, audio, animations, three dimensional images and animations, virtual reality environments, text, text-to-speech conversion, animated characters or "agents", and web pages. Additionally, the plugins preferably include plugins for handling interaction, such as a plugin for connecting to the Internet or other two-way communication channel, and communicating with a server over the Internet or other two-way communication channel (e.g. for placing bids in an

auction show, or participating in chat or game shows), and for permitting the user to enter data (through a keyboard, mouse, voice, etc.).

Once webshow engine 8 has activated its three subcomponents, player control 11 of client 7 in turn instantiates webshow plugin collection manager 13, which is used for managing all of the webshow plugins to be accessed by the flow of the webshow. After webshow plugin collection manager 13 is instantiated, client 7 is in a ready state, able to accept calls from outside computer components. At this point, webshow engine 8 can request from data control 9 a particular webshow.

Webshows are stored in what are referred to as index files. If a particular webshow requested by webshow engine 8 is stored locally in data resource manager 12, then data control manager 9 can send that local index file 14 to webshow engine 8 for processing. However, if the requested index file is not located in data resource manager 12, then data control manager 9 has to request the webshow from a server which contains webshow index files 22.

FIG. 5 is a detailed depiction of webshow index file 22. Each webshow index file 22 has five meta-data properties: station, show, date, time, and description. Each of the meta-data properties are defined in the header of webshow index file 22. In addition to the meta-data properties, each webshow index file 22 also contains two subcomponents: webshow plugin group 23, and webshow content group 24.

After downloading the webshow index file 22, webshow engine 8 then examines webshow index file 22 to ensure that it is a properly formatted webshow index file. If the webshow index file is properly formatted,

the web show engine 8 passes the contents of webshow index file 22 to index manager 10.

Once index manager 10 receives webshow index file 22 from webshow engine 8, index manager 10 creates a local index file 14 that is named with the URL from which webshow index file 22 originated. Index manager 10 then passes the contents of webshow index file 22 to the newly created local index file 14.

Local index file 14 first takes webshow plugin group data 23 and passes that data (via webshow engine 8) to player control 11, and then to webshow plugin collection manager 13. Webshow plugin group 23 contains multiple webshow plugin links 25. Each webshow plugin link 25 refers to an external software component that manages some aspect of a webshow. Webshow plugin links 25 contain a form of identification by which the particular plugin link will be referenced ("ID"), and a URL containing the location of the plugin ("SRC").

Once the webshow plugin collection manager 13 retrieves the webshow plugin group data, webshow plugin collection manager 13 creates a webshow plugin collection 15 with the same name as the originating index file. Webshow plugin collection manager 13 then passes the webshow plugin group data to newly created webshow plugin collection 15. Webshow plugin collection 15 in turn creates a webshow plugin 16 for every plugin link. Each webshow plugin 16 instantiates the software component as indicated by the SRC attribute for each webshow plugin.

Once this first level of parsing local index file 14 is completed, local index file 14 activates content manager 17, and passes to it the data that was contained within webshow content group 24. Each

webshow content group 24 may contain a plurality of webshow content files 26. For each of webshow content file 26, content manager 17 creates corresponding local content file 18. Each local content file 18 contains an ID by which it can be identified, and an SRC attribute, which contains the location of the actual contents of webshow content file 26. Local content file 18, upon obtaining the SRC attribute, will check for the presence of the file indicated by the SRC attribute in data received via broadcast, or will download the file indicated by the contents of the SRC attribute. Local content file 18 will then ensure that the transmitted or downloaded file is properly formatted. If the downloaded file is properly formatted, then local content file 23 will extract every webshow instruction 28 from the downloaded or transmitted webshow content file 26, and create a corresponding local instruction 19 for local content file 18.

Every webshow instruction 28 is the child of a webshow plugin directive 27. Therefore, each local instruction 26 is given a plugin attribute (which is identical to an ID attribute (not shown) of webshow plugin directive 27), so that each local instruction 19 can be easily routed to the appropriate webshow plugin 16 for processing.

Each webshow instruction 28 may contain a plurality of webshow parameters 29, depending upon the function performed by the plugin. The data contained within webshow parameters 29 in webshow instruction 28 is passed into corresponding local instruction 19 in local content file 18, at which time name and value attributes are assigned to each local parameter 20 contained within local instruction 19.

Once the contents of webshow content file 26 have been parsed in this manner, local index file 14 instantiates instruction manager 21. Instruction manager 21 searches through all local content files 18 in local index file 14, creating a list of all the instructions contained within the various local content files 18. Creating such a list of all instructions contained within all of the content files provides for ease in adding, moving, or modifying instructions at run-time.

At this point, the data, flow, and display of the webshow have now been separated, and webshow engine 8 indicates to the user that the user can now play the loaded webshow. The user can then play the now-loaded webshow through scripting or some other similar method. Once the user requests that webshow engine 8 play the currently-loaded webshow, webshow engine 8 passes that request to player control 11. Player control 11 then asks instruction manager 21 for the current local instruction. When player control 11 gets a current local instruction 19, it passes that local instruction 19 to the appropriate webshow plugin 16 by matching the ID attribute of local index file 14 with the ID of webshow plugin collection 15. When these IDs are matched, then local instruction 19 gets passed into webshow plugin collection 15, where webshow plugin collection 15 matches the local instruction's plugin attribute with the appropriate webshow plugin's SRC attribute. When this match has been made, local instruction 19 gets passed to webshow plugin 16, which in turn passes local instruction 19 to the external software component associated with the particular webshow plugin 16 for processing. When the external software component has finished processing the local

instruction 19 and signals to the webshow plugin 16 that it is finished, then the webshow plugin 16 signals player control 11 to go to the next instruction, whereby this process repeats until webshow engine 8 receives a request to modify the flow of the webshow, or the webshow reaches the end of the collection of local instructions.

The flow of a webshow can be modified in a number of ways. First, a webshow can be stopped. When the webshow engine receives the stop command (e.g., from the server or client), webshow engine 8 notifies the player control to send all the webshow plugins 20 a stop message. When the stop message reaches the webshow plugin 20, it in turn sends a stop commend to the associated external software component. At this point, the external software component stops processing the instruction

Another flow altering command is the fast forward command. The fast forward command is similar to the stop command, except once the external software component stops processing the instruction, player control 11 requests that instruction manager 22 move to the next instruction. If webshow engine 8 was stopped, nothing occurs; however, if webshow engine 8 was playing, then it gets the next instruction and continues with the play sequence. If the current instruction is the end of the instruction collection, and there is a fast forward request, then the current instruction loops around to the first instruction in the instruction collection.

A third flow altering command is the rewind command. The rewind command operates in the same manner as the fast forward command except, instead of requesting that instruction manager 22 move forward one

instruction, it requests that instruction manager 22 move back one instruction. If the current instruction is the first instruction in the instruction collection, and there is a rewind request, then the current instruction loops around to the final instruction in the instruction collection.

Another flow altering command is the goto command, which is usually generated by an external software component associated with a particular plugin. This command results in the flow being stopped (as previously described in the stop sequence above). Then, player control 11 requests that index manager 10 move to a particular instruction. At that point, if the webshow was previously playing, a play command will be issued; otherwise, the webshow will remain stopped.

Moreover, the fast forward, rewind, and goto commands can be used to move through bookmarks, as opposed to instructions. In a preferred embodiment, these bookmarks may correspond to thematic units of the webshow, such as a play, act, or scene.

Finally, the flow may be altered by changing local content files 18 while in the middle of playing the webshow. Content files 18 generally contain a scripted series of events. Since this scripted series of events is accessible to the user via webshow engine 8 (and through content manager 17), the user can change these individual events during play, as long as the content file to be changed is not the content file being broadcast at that moment. Similar changes may be made from the server by a broadcaster or business.

This entire process can be repeated with any number of local index files 17, by loading more local index files 17 as described above. However, since each local index file 17 corresponds to one webshow, only

one local index file 17 can be active at any one time; all other loaded index files must be in the stop state.

As described above, separating telecast content into DEBOs, namely data 1, flow 2, and display methods 3, and then using different controls to manage each of these DEBOs provides for a high degree of interactivity and reusability. By using player control 11 to manage the display methods, as opposed to playing media hard coded to a web page, client 7 creates its own interface to the plugins that are actually used to display the telecast content. Thus, webshow engine 8 can send whatever media the user chooses to the particular plugin on a real-time basis. Likewise, by using index manager 10 to manage the flow of the telecast content, client 7 can make changes to the scripted series of events that comprise the webshow, regardless of media type (e.g., data 1) or media player (e.g., display 3), and can do so as the webshow is running. Finally, by using data control manager 9 to manage the data objects of telecast content, client 7 can alter or replace data 1 seamlessly, because data 1 now has properties that describe what the data is used for, as opposed to simply being tied to a document or cached in a web browser.

At present, a preferred embodiment of client 7 comprises a personal computer connected to the Internet, running Microsoft Windows 98, and the Internet Explorer 5 web browser. As will be understood by one skilled in the art, client 7 may comprise most any device capable of receiving telecast content, and of being programmed to handle the control of flow, data, and display required by the present invention, and does not require a web browser or any particular operating system. Platforms on which client 7 could be

advantageously installed include personal computers, set-top boxes, media appliances, and televisions containing specialized electronics to handle the system of the present invention.

Referring now to FIG. 6, the structure of a server built in accordance with the principles of the present invention is shown. Broadcast portal server 30 is a server, located at a central site, which stores and distributes telecast content, or webshows, to multiple clients. Broadcast portal server 30 uses three sub-components to transact a complete webshow: store transactions service 31, distribute transaction services 32, and calculate transaction services 33.

Generally, store transaction service 31 contains all of the files containing data, flow, and display information that will be sent to clients 7, and all of the data needed by the transaction services to distribute telecasts to clients. Store transaction service 31 also holds scheduling information, and may include various administrative data, such as demographic data on the current viewers, and accounting data on which advertisements have been displayed to various audience groups.

Store transaction service 31 consists of three primary database services 34: SQL database 35, Microsoft Transaction Service (MTS) 36, and object oriented (XML) database 37. SQL database 35 contains stored procedures 38 that create or confirm individual file identifications. Thus, all user information is stored in SQL database 35.

XML object oriented database 37 stores objects encoded in WSML. The primary WSML component stored in XML object oriented database 37 is WSML cartridge 41. WSML cartridge 41 is the sum of all

index files 14 and associated data files used to provide telecast content to clients 7.

MTS 36 is the object bridge that allows broadcast portal server 30 to maintain a connection to thousands of clients and to retrieve data stored in the SQL database in real time. While MTS is used in a preferred embodiment of the invention, one of ordinary skill in the art would recognize that any other industry standard object bridge could be used in place of MTS 36.

Distribute transaction service 32 handles the routing of data between store transaction service 31 and calculate transaction service 33, and between the different levels of station, program, and room. The major software component of distribute transaction service 32 is echo TV server 46. Echo TV server 46 contains TV station server 47, TV Program server 48, and TV room server 49. These servers determine the "who", "where", "when", and "in what order" to distribute webshow instructions 28. These distribute functions are derived from WSML cartridge 41.

Each of TV station server 47, TV program server 48, and TV room server 49 contain TV data stream router 51 and communication controller 52, which are server components that help the system communicate between the calculate transaction service 33 and the store transaction service 31. The server components also enable the server at either the level above or the level below the current level to make changes in real time.

Moreover, distribute transaction services 32 are used to initialize broadcast portal server 30. Specifically, to initialize broadcast portal server 30, WSML cartridge 41 must be uploaded or dragged and

dropped into broadcast portal I-initializer 60. I-intializer 60 then parses WSML cartridge 41 into webshow index files 22, and parses webshow index files 22 into their component files and component data files. I-initializer 60 then takes the parsed component files and sends commands to portal creator 59. Portal creator 59 then instantiates the correct objects and directs them to the correct subset level (station, room, or show) of chat engine 57 (within chat engine 57 are already station levels, webshow levels, TV room levels, and seat enumeration).

Calculate transaction services 33 primarily consists of TV station plugin servers 64, TV room plugin servers 66, and TV program plugin server 65. These servers are referred together as transaction logic plugin servers 63. Each transaction logic plugin server 63 contains the following servers: procedure plugin server 67, which decides if data needs to be routed up or down to transaction servers other than the one to which it is attached; TV plugin commerce server 68, which handles all transactions for that particular server; TV logic plugin server 69, which both handles webshow variables and webshow instructions and contains the actual logic for a particular show, such as an auction or a news program; and extensible natural language processing (XNLP) plugin server 70, which parses natural chat messages from clients and makes decisions on how to act based on natural language processing. For example, XNLP plugin server 70 could be used to react to the use of profanity within a chat room, or could be used to change the types of goods being auctioned on an auctioned program based upon participants' reactions.

Another way to delineate these three services on broadcast portal server 30 discussed above is to examine and understand their demands on resources. Store transaction service 31 requires extensive hard drive access and space. Distribute transaction service 32 requires expansive random access memory. Calculate transaction services 33 require intensive processor resources. By separating broadcast portal server 30 into different resource services, the administration of broadcast portal server 30 can be distributed over a larger server farm, and as a result can handle millions of concurrent users. Or, on the other hand, if the system is only required to handle a few dozen users, the entire system can run on one robust server.

Referring to FIG. 6 in conjunction with FIG. 7, the connection between client applications 71 and broadcast portal server 30 is shown. Broadcast portal server 30 can deliver telecasts to various kinds of client applications 71. Each client application 71 is able to receive data through a standard analog or digital broadcast signal 82, as well as through a two way communication channel, such as the Internet, which uses a TCP/IP connection protocol. Because of the work of the ATVEF group, described above, most, if not all client applications 71 will be able to extract data using ATVEF drivers from the broadcast signal. The system described herein does not interfere with the ATVEF standard, but rather extends it.

Further, because echo TV sockets 56 are made for TCP/IP connections, it is assumed that all client applications 71 have a TCP/IP layer built in order to connect to broadcast portal server 30. It will be understood by one skilled in the art that any network

or communication protocol could be used in place of TCP/IP without departing from the present invention.

There are three types of client applications 71 with which broadcast portal system 30 system is designed to interface. The first is the set-top client 72. Set-top client 72 is either a set-top box or an Internet enabled appliance. Because of the limitations of set-top client 72, web server 78 directs set-top client 72 via active server pages (ASP) 79 or HTML link re-routes. This re-routing will be accomplished on the server by way of a browser sniffer. The browser sniffer is contained within ASP 79 script files or hypertext application (HTA) page 84 contained within web server 80. The HTML will initialize and download the correct client software so that set-top client 72 can connect to echo TV server sockets 56 within echo TV server 46. This allows a user to gain access to the broadcast portal service 30 described herein.

The second type of client application 71 is webshow administration client 73. Since webshow administration client 73 connects via a built-in initialization script, it must connect to the correct echo TV server socket 56. Webshow administration client 73 performs several important administrative functions. When broadcast portal service 30 is not yet initialized, webshow administration client 73 can file transfer WSML cartridge 41 to broadcast portal service 30, which would then parse WSML cartridge 41 via I-initializer 60. I-initializer 60 then creates the appropriate TV station server 47, TV program server 48, and TV room server 49 so that the administrator can operate broadcast portal server 30.

Next, webshow administration client 73 allows the administrator of broadcast portal server 30 to

modify webshow cartridge 41. The administrator can use webshow administration client 73 to access XML database 37 and download WSML cartridge 41. WSML cartridge 41 can then be edited within any standard XML editor, and can be saved back to the XML database 37, or be file transferred directly to broadcast portal server 28.

The third type of client application 71 is PC-TV client 74. PC-TV client 74 will be able to initialize and connect to the correct echo TV server socket 56 via the correct HTA file.

Although these embodiments describe the use of ASP and HTA files to simplify connections over the Internet, one of ordinary skill in the art would recognize that other means of forming connections over the Internet other than ASP and HTA are available, and may be used without departing from the invention.

Once a client application 71 has connected and been confirmed with echo TV server socket layer 56, the system will perform a set of SQL stored procedures 38, connecting through store transaction service 31 to SQL database 35. These stored procedures create (for first time users of the broadcast portal server 30) or confirm (for repeat users) individual file identifications. Once identified, the system will pass the user through to the correct service on broadcast portal server 30. As mentioned above, the use of MTS allows echo TV server 46 to maintain connections with thousands of clients, and marshal and maintain database services 34 while databases 35 and 37 are in process. Echo TV server 46 can transact with MTS.

At this point, the user is confirmed and the identification requirements have been satisfied. However, before echo TV server 46 can proceed, it needs feedback from the user to determine what telecast

content to provide the user. Thus, echo TV server 46 will parse basic chat instructions provided by the user by means of gateway 61. Once the appropriate feedback has been received, client application 71, through a WSML instruction 28, will request a specific station, webshow, and in some cases a specific room for the user.

Referring now to FIG. 8, the process by which a WSML cartridge 41 interacts with distribute transaction services 32 in order to provide telecast content to client 7 is shown. WSML cartridge 41 contains four index files: web show index file 14, TV station index file 86, TV program index file 87, and TV room index file 88. Each index file 14, 86, 87, and 88 contained within WSML cartridge 41 contains all the correct station, webshow, TV room, and appropriate transaction logic to run interactive broadcast portal server 30. Each of these index files also contains a corresponding component file (16, 89, 90, and 91, respectively).

Therefore, despite the fact that WSML cartridge 41 contains all the index files, and all corresponding component files, each level (station, program, or level) of index file does not contain the entire WSML cartridge 41. I-initializer 60 breaks WSML cartridge 41 into the correct sub-cartridges for station, room, and show, and instantiates the appropriate distribute transaction services 32 and transaction logic plugin services 63 for that level. For example, TV station server 47 contains its own sub-set of WSML cartridge 41 in its own memory space. Only the sub cartridge of WSML cartridge 41 that pertains to the particular TV service (in this example, TV station

server 47 and TV station plugin server 64)is held in the memory space of the specific server.

Further, WSML cartridge 41 only instantiates the transaction logic plugin services 63 that are required to run the particular webshow. WSML cartridge 41 only requires transaction logic plugin services 63 when a user or group "transaction" will occur, where a transaction is any communication involving more than one person that affects all those involved. For example, chat or hosting a chat session, an auction, and multi-player games are all transactions. Because transactions affect more than a single user, all transactions are the domain of the broadcast portal server 30. However, client 7 allows a user to personalize a TV show with or without transactions.

The WSML cartridge subsets are composed of XML branched instruction sets. Each instruction set can be further categorized into its own subsets of play, act, and scene. Within each of these subsets of play, act, and scene are a set of WSML webshow instructions 28. These webshow instructions 28 tell the distribute transaction services 32 how, when, and why to distribute data. These instructions create the flow of the webshow that is required to be followed in the course of any interactive broadcast.

Referring to FIG. 9, each of these webshow instructions 28 will create possible message sets. Message sets are either webshow instructions 28, webshow events 98, or webshow variables 99, or webshow chat 100. These message sets can then be delivered to a different level of TV services (either station, program, or room), to its own level or another level of plugin server, or to store transaction services 31 or client 7.

Webshow instructions 28, as discussed above, are a set of instructions encoded in WSML that are inserted either into one of transaction logic plugin servers 63, or webshow engine 8. The TV station server 47, TV program server 48, and TV room server 49 all contain a webshow engine for parsing the WSML language. As a result, client 7 has the ability to run all functions (because it contains webshow engine 8), so that client 7 can maintain a connected state to a one-way stream. In this scenario, webshow instructions 28 and webshow variables 99 are sent directly to client 7 via the broadcast data stream or TV signal. Non-connected clients are synchronized with connected clients via this broadcast stream. Thus, all clients, whether connected or not-connected, receive transactional data; the difference is that only connected clients can actually send or add transactional data to the server in real-time. However, even non-connected clients can input data, since client 7 is self contained: client 7 can simply store all transactions in data resource manager 12 until connected. Yet where collaboration, competition, or chat are required between clients 7, dynamic events need to be calculated by the server (and not the client) so that all clients 7 can be synchronized.

Referring back to FIG. 8, after an individual user has been registered, the system places the user in a specific TV room, which is a subset of a specific TV show, which is, in turn, a subset of a specific TV station. The TV room index file initializes all TV room plugin servers 66 and TV room server 49. TV room engine 55 (a component of TV room server 49) holds in memory both TV room index file 88 and its corresponding

TV room component file 91. TV room engine 55 is what actually sequences WSML webshow instructions 28.

The TV room can have various parameters. Typical parameters for a TV room include a total number of individuals within a certain demographic, such as a postal code. The TV room can change as needed, or parameters can change based upon routines from TV program server 48 or TV station server 47, or even webshow administration client 73. All parameters are stored in the memory of TV room procedure plugin server 67. TV room plugin server 67 determines whether or not a user should be in a room, or if a new room should be created and the user sent to that room.

Communication controller 52 handles communications between transaction logic plugin servers 63 and TV station, program, and room servers 47, 48, and 49. Communication controller 52 also enables the TV station, room, and program servers to communicate with any of webshow engine 8, procedure plugin servers 67, TV plugin commerce servers 68, TV logic plugin servers 69, or XNLP plugin servers 70. While in a preferred embodiment the COM interface is used to govern these communications, one of ordinary skill in the art would recognize that other communication interfaces could be used.

XNLP plugin servers 70 are natural language processors that handle queries from the user and user group and respond by communicating directly through chat engine 57, or by sending specific commands to TV room engine 55. For example, a user in a TV room of a city guide webshow may ask where they could find a good sushi restaurant. XNLP processor 70 would respond by first processing the chat string via a language query. In this example, XNLP processor 70 would recognize the

terms "restaurant" and "sushi". Next, XNLP processor 70 would send a command to MTS 36, which would then retrieve the specific information from XML database 37. This information would then be directed through TV data stream router 51 back to echo TV server 46 and to all clients 7 in that TV room. Such natural language plugins are previously known and commercially available.

The function of MTS 36 is to make connections to databases 35 and 37, and update changes to TV room index file 88 and TV room component file 91. It also makes changes to particular object variables, whose value may have changed during the process of the show.

TV data stream router 51 receives its data from either MTS 36, UDP stream media server 81, or from TV room engine 55. TV data stream router 51 handles all communications outside of the TV room domain. Only TV data stream router 51 can communicate to a level below or above the TV room domain. Communications controller 52, on the other hand, handles communications within the TV room domain. For example, communications controller 52 allows communications with various TV room plugin servers 67.

At present, a preferred embodiment of broadcast portal server 30 comprises an Intel-processor-based server, running Microsoft Windows NT 4.0, or Linux, as well as software performing the tasks described hereinabove. As will be apparent to one skilled in the relevant arts, other platforms may perform the same functions, if provided with software built in accordance with the principles of the present invention.

Referring now to FIG. 10, a preferred embodiment of the webshow user interface screen is

shown. The webshow user interface screen displays a variety of objects. Each object is handled by a specific webshow plugin. Moreover, webshow players that comprise the user screen can handle any kind of media, including standard types such as mpeg 1, mpeg 2, mpeg 4, Macromedia's Shockwave, VRML, DHTML, Microsoft's Agent characters; and standard image formats such as JPG, GIF, and PNG file types. Further, the system allows multiple players and multiple objects to be layered upon one another.

The first set of plugins are navigation buttons 102. These buttons enable the user to specify topics of interest within a specific webshow, and then move the show forward and backward to the particular topic selected by the user depending on how and where those topics exist within the specific format of the webshow.

The second set of webshow plugins is Alpha Channel video sprite 103 with digital video stream, which stream is made transparent and layered onto the TV screen. This control can layer any size or shape video layer onto the screen. The player also can handle audio and video.

Another set of plugins that could be used includes advertising banner 104, which simply rotates advertisements based on commands from broadcast portal 30.

Webshow controls 105 are the main controls of the webshow or broadcast portal service 30. These controls enable the user to stop, play, rewind, or fast forward. Webshow controls 105 allow the user to stop all events, media, and media players on the screen. They also allow the user to fast forward to events that have not played if the media required is accessible.

Media is made accessible either by client 7 itself, UDP media stream server 81, or the broadcast signal stream. HTTP can be employed when specific media files are required. Webshow controls 105 also enable the user to skip from bookmark to bookmark, which can, for example, correspond to segments, acts, or scenes.

Other webshow players 113 include: webshow active background 106, which is a container of any media type, including video animation or 3-D environment; main media window 107, which handles all types of video, audio, 3-D, or animation file types, including streaming media such as Microsoft's and Real Network's streaming video formats; and 3-D TV anchor services 109, which include an animated TV anchor 110. 3-D TV anchor services 109 include voice recognition, text to speech engine, and advanced animation features. The TV anchor can handle multitasking and various unscripted tasks, which the webshow handles in the course of its interactive functions. For example, during a travel show the TV anchor could answer questions from the user about specific places to visit in New York City.

The webshow player interface also includes user input plugin 108, which handles any user input. Typically, user input plugin 108 handles either text strings or text variables as formatted by the graphical user interface.

Referring now to FIG. 11, the method by which a user is able to generate and store dynamic WSML cartridges is described. A user is able to create a dynamic WSML cartridge, which consists of a WSML cartridge that has been customized by the user to contain particular webshow media (such as video, animation, or audio files) that has been stored with

the WSML structure (i.e. separating the data, flow, and display) on data resource manager 12, or in database services 34. Datatube control 113 can then play the stored content. This customized media is then categorized into sets of datatube cartridges.

As mentioned above, client 7 can operate without a server side connection: it simply plays the stored datatubes based upon user preferences. However, client 7 can also be connected to a server and updated. Thus, the datatube can be updated to play new media based upon an old or familiar format. For example, a "sport news center" format can have the media switched and deliver cooking recipes in a sports news format and style. Alternatively, the format can change, while the media stays the same. In the proposed example, the latest sports statistics could be presented in the manner of a cooking show.

Datatube control 113 can contain references to the software required to alter the WSML cartridge 41. Thus, the TV show stored as a datatube has the software built inside of it that is necessary to run the datatube.

Datatube control 113 has six elements which enable it to replay a stored webshow as well as update those shows dynamically at runtime. In a typical scenario, webshow engine 8 would request a saved webshow from datatube registry 114. The registry would check to see if the requested webshow was available in the data control 2. It would also begin a set of variable retrievals from user ID base 118. User ID base 118 verifies the correct settings and preferences for the requested object. The requested object then gets a new pointer from webshow linked list 119, which has stored within it all the learned personal

preferences of all the users of the system. The requested object is now passed to webshow resource manager 117, which interprets the request, and then retrieves the correct WSML cartridge 41 from data control 2. Webshow resource manager 117 then verifies all needed components, and passes WSML cartridge 41 to active TV component 121. Active TV component 121 parses all the WSM instructions that it has requested from webshow linked list 119. Webshow linked list 119 has stored all webshow variables 99 and webshow instructions 28 it created during the running of previous webshows. WSML cartridge 41 only has a set of datatube-only WSML instructions. These instructions are stored as WSM (web show macro) files 125, which are added from the webshow linked list in active TV component 121. The enhanced dynamic WSML cartridges are passed to datatube registry 114 for updating the registry component. Datatube registry 114 then passes the newly registered dynamic WSML cartridge to active TV library 115. Active TV library 115 runs all the WSM instructions in WSM files 125. The result is a newly formed WSML cartridge 41 amended to reflect the dynamic or personal preferences of the user. Webshow engine 8 is unaware of these changes because it does not parse, send, or receive the datatube control's WSML instructions 28. Rather, active TV library 115 strips the commands or instructions 119 once all WSM instructions are complete.

As a result, client 7 has the ability to play a TV show as it was broadcast or the user can change some parts of the broadcast, either by dynamically updating the content, or by interchanging the data, flow, or display functions or webshow events 98. Datatube control 113 handles all the changes or

alterations to stored WSML webshow instructions 28, or events 98.

Although preferred illustrative embodiments of the present invention are described above, it will be evident to one skilled in the art that various changes and modifications may be made without departing from the invention. It is intended in this application will cover all such changes and modifications that fall within the true spirit and scope of the invention.

What is claimed is:

1. A method of providing a personalized interactive telecast, the method comprising:

separating portions of the telecast, and packaging the separated portions as a parsed datastream comprising:

a data portion, that contains a plurality of objects to be used in the telecast;

a flow portion, that describes a sequence in which the plurality of objects are used; and

a display method portion, that describes, for each type of object in the plurality of objects, a method for using that type of object;

continuously sending the parsed datastream over a broadcast medium from a server to a client;

altering one or more of the data portion, the flow portion, or the display method portion on the client to personalize the telecast; and

displaying the telecast on the client by using the content of the data portion in an order specified by the flow portion, using the methods specified in the display method portion.

2. The method of claim 1, wherein displaying the telecast on the client by using the content of the data portion in an order specified by the flow portion, using the methods specified in the display methods portion further comprises:

layering multiple objects on a display using the display methods for the objects being layered on the display; and

replacing layers of objects while the interactive telecast is being displayed.

3. A method for displaying a personalized interactive telecast on a client, the method comprising:

receiving a user's request to view a particular telecast;

retrieving telecast content from an object oriented database based upon information provided from said user;

separating portions of said telecast content, and packaging the separated portions as a parsed datastream comprising:

a data portion, that contains a plurality of objects to be used in the telecast;

a flow portion, that uses flow objects to describe a sequence in which the plurality of objects are used; and

a display method portion, that describes, for each type of object in the plurality of objects, a display method for using that type of object;

storing said data portion of said parsed datastream in an object oriented database;

ordering said flow portion of said parsed datastream into a series of instructions executable by said display methods;

executing said instructions in a sequential order indicated by the flow objects of said flow portion by routing each object in said plurality of objects contained in said data portion to an appropriate display method for that object.

4. The method of claim 3, wherein executing said instructions in a sequential order indicated by the flow objects of said flow portion further comprises:

altering one or more of said data portion, said flow portion, or said display method portion on the client to personalize the telecast while the instructions are being executed.

5. The method of claim 4, wherein altering one or more of said data portion, said flow portion, or said display method portion on the client to personalize the telecast comprises altering one or more of said data portion, said flow portion, or said display method portion on the client to personalize the telecast while the client is not maintaining a two-way connection to a server.

6. The method of claim 3, wherein ordering said flow portion of said parsed datastream into a series of instructions executable by said display methods further comprises:

altering one or more of said data portion, said flow portion, or said display method portion on the client to personalize the telecast prior to executing said instructions.

7. The method of claim 3, wherein retrieving telecast content comprises:

searching for the requested telecast in an object oriented database located at the client;

verifying the integrity of said telecast content; and

retrieving any data found to be missing from said telecast content from an object oriented database located at a centralized server.

8. A method for delivering distributed entertainment broadcast objects as a personalized interactive telecast from a server, the method comprising:

verifying a user's identification based upon a set of stored procedures;

retrieving telecast content from an object oriented database based upon information provided by said user;

separating portions of said telecast content, and packaging the separated portions as a parsed datastream comprising:

a data portion, that contains a plurality of objects to be used in the telecast;

a flow portion, that describes a sequence in which the plurality of objects are used; and

a display method portion, that describes, for each type of object in the plurality of objects, a method for using that type of object; and

delivering said parsed datastream to said user.

9. The method of claim 8, further comprising:

separating the distributed entertainment broadcast objects into:

a store portion that contains a second plurality of objects to be used by the server;

a distribute portion that describes routes, sequences, and schedules on which the second plurality of objects are used; and

a calculate portion, that describes, for each type of object in the plurality of objects, a logic for using that type of object.

10. The method of claim 8, wherein delivering said parsed datastream to said user comprises delivering said parsed datastream to said user over a broadcast medium.

11. The method of claim 8, wherein delivering said parsed datastream to said user comprises delivering said parsed datastream to said user over the Internet.

12. The method of claim 8, wherein delivering said parsed datastream to said user comprises delivering said parsed datastream to said user over a two-way communication channel.

13. The method of claim 8, further comprising:

altering one or more of said data portion, said flow portion, or said display method portion at the server to personalize the telecast while said parsed datastream is being delivered to said user.

14. The method of claim 8, wherein retrieving telecast content from an object oriented database based upon information provided by said user further includes:

parsing natural language instructions provided by said user with a natural language processor to produce parsed natural language instructions; and

identifying telecast content based upon the parsed natural language instructions.

15. The method of claim 8, wherein separating portions of said telecast content, and packaging the separated portions as a parsed datastream further comprises:

opening a room level transaction engine for handling transactions at a room level;

opening a show level transaction engine for handling transactions at a show level, the show level transaction engine associated with one or more room level transaction engines;

opening a station level transaction engine for handling transactions at a station level, the station level transaction engine associated with one or more show level transaction engines;

placing the user in a particular show at the show level based upon input from the user; and

placing the user in a particular room at the room level according to demographics of the user.

16. The method of claim 15, further comprising selecting logic for each of the room transaction engine, the show transaction engine, and the station transaction engine based on input from the user.

17. The method of claim 15, further comprising selecting logic for each of the room transaction engine, the show transaction engine, and the station transaction engine based upon the business needs of a business.

18. A method for storing interactive media content, the method comprising:

separating portions of the telecast content, and packaging the separated portions as a parsed datastream comprising:

a data portion, that contains a plurality of objects to be used in the telecast;

a flow portion, that describes a sequence in which the plurality of objects are used; and

a display method portion, that describes, for each type of object in the plurality of objects, a method for using that type of object;

altering portions of said parsed datastream based upon a set of preferences input by a user to produce an altered parsed datastream;

storing said altered parsed datastream in a database;

updating the plurality of objects of the data portion of said altered parsed datastream; and

displaying said altered parsed datastream with updated content in accordance with said user preferences.

19. A client device that displays a personalized interactive telecast, the client device comprising:

a display; and

an object oriented database that stores telecast content;

wherein the client device is programmed to:

retrieve telecast content from the object oriented database based upon information provided by a user;

separate portions of said telecast content, and package the separated portions as a parsed datastream comprising:

a data portion, that contains a plurality of objects to be used in the telecast;

a flow portion, that describes a sequence in which the plurality of objects are used; and

a display method portion, that describes, for each type of object in the plurality of objects, a display method for using that type of object;

store said data portion of said parsed datastream in the object oriented database;

order said flow portion of said parsed datastream into a series of instructions executable by said display methods; and

execute said instructions in a sequential order indicated by the flow objects of said flow portion by routing each object in said plurality of objects contained in said data portion to an appropriate display method for that object, to show a personalized interactive telecast on the display.

20. The client device of claim 19, wherein the client device receives telecast content from a server via a broadcast medium.

21. The client device of claim 19, wherein the client device receives telecast content from a server over the Internet.

22. The client device of claim 19, wherein the client device receives telecast content from a server over a two-way communication channel.

23. The client device of claim 19, wherein the client device comprises a personal computer.

24. The client device of claim 19, wherein the client device comprises a set-top box and the display comprises a television.

25. The client device of claim 16, wherein the client device comprises a media appliance.

26. The client device of claim 16, wherein the client device comprises an advanced television set.

27. A server that delivers distributed entertainment broadcast objects as a personalized interactive telecast to one or more clients, the server comprising a computer having an object oriented database, the server programmed to:

retrieve telecast content from the object oriented database based upon information provided by a user;

separate portions of said telecast content, and package the separated portions as a parsed datastream comprising:

a data portion, that contains a plurality of objects to be used in the telecast;

a flow portion, that describes a sequence in which the plurality of objects are used; and

a display method portion, that describes, for each type of object in the plurality of objects, a method for using that type of object; and

deliver said parsed datastream to the one or more clients.

28. The server of claim 27, wherein the server is programmed to deliver said parsed datastream to the one or more clients via a broadcast medium.

29. The server of claim 27, wherein the server is programmed to deliver said parsed datastream to the one or more clients over the Internet.

SUBSTITUTE SHEET (RULE 26)

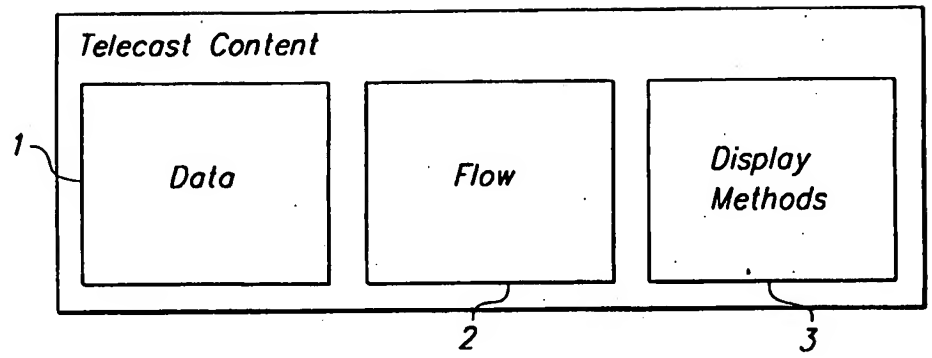


FIG. 1

```
<WSC Show="SportsShow">
<Anouncer1>Costas.Anchor</Announcer>                    1
<Announcer2>Somerall.Anchor</Announcer2>
<Dialogue>
    <Intro1> Welcome to Sports Center, tonight is going to be a great show!</Intro1>
    <Intro2> Wow this is going to be one amazing Sports Center show!</Intro2>
</Dialogue>
<Image>
    <AnnouncerGifs>
        <Gifs1>Costas.gif</Gifs1>
        <Gifs2>Somerall.gif</Gifs2>
    </AnnouncerGifs>
</Image>
<Sound>
    <Music>SportsCenter.wav</Music>
</Sound>
...
</WSD>
```

FIG. 2A

```
<WSC Show="SportsShow">                                    2
<Load>Announcer1</Load>
      <Display>Image/AnnouncerGifs/Gifs1</Display>
      <Action>Speak Intro1</Action>
<Load>Announcer2</Load>
      <Display>Image/AnnouncerGifs/Gifs2</Display>
      <Action>Speak Intro2</Action>
...
</WSC>
```

FIG. 2B

```
<WSI Show="SportsShow">                                    3
<PlugIn id="Video_Player"
      MacSrc="QuickTime"
      PcSrc = "NetShow"
...>
...
</WSI>
```

FIG. 2C

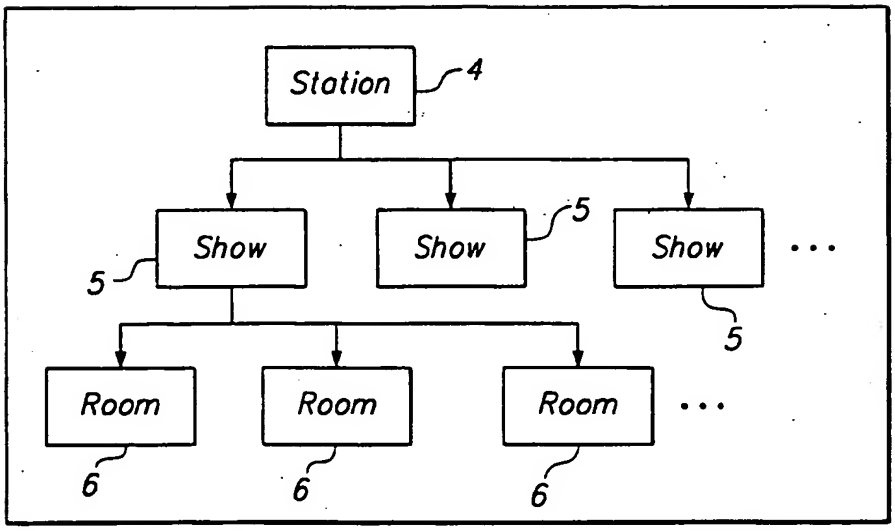


FIG. 3

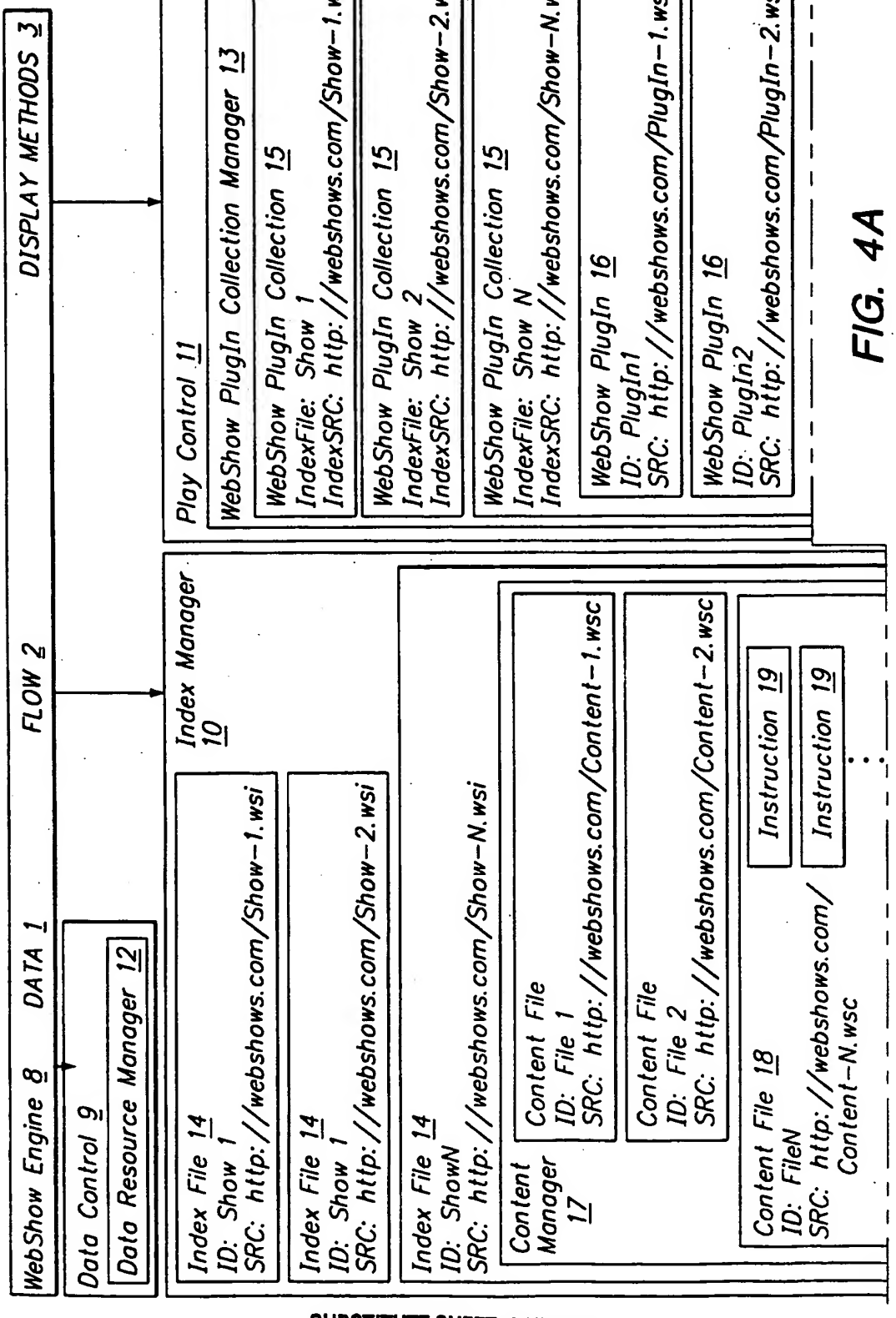

WebShow Engine 8
DATA 1
FLOW 2
DISPLAY METHODS 3
Data Control 9
Data Resource Manager 12
Index Manager 10
Index File 14
ID: Show 1
SRC: http://webshows.com/Show-1.wsi
Index File 14
ID: Show 1
SRC: http://webshows.com/Show-2.wsi
Index File 14
ID: ShowN
SRC: http://webshows.com/Show-N.wsi
Content Manager 17
Content File
ID: File 1
SRC: http://webshows.com/Content-1.wsc
Content File
ID: File 2
SRC: http://webshows.com/Content-2.wsc
Content File 18
ID: FileN
SRC: http://webshows.com/Content-N.wsc
Instruction 19
Instruction 19
Play Control 11
WebShow PlugIn Collection Manager 13
WebShow PlugIn Collection 15
IndexFile: Show 1
IndexSRC: http://webshows.com/Show-1.wsi
WebShow PlugIn Collection 15
IndexFile: Show 2
IndexSRC: http://webshows.com/Show-2.wsi
WebShow PlugIn Collection 15
IndexFile: Show N
IndexSRC: http://webshows.com/Show-N.wsi
WebShow PlugIn 16
ID: PlugIn1
SRC: http://webshows.com/PlugIn-1.wsp
WebShow PlugIn 16
ID: PlugIn2
SRC: http://webshows.com/PlugIn-2.wsp

FIG. 4A

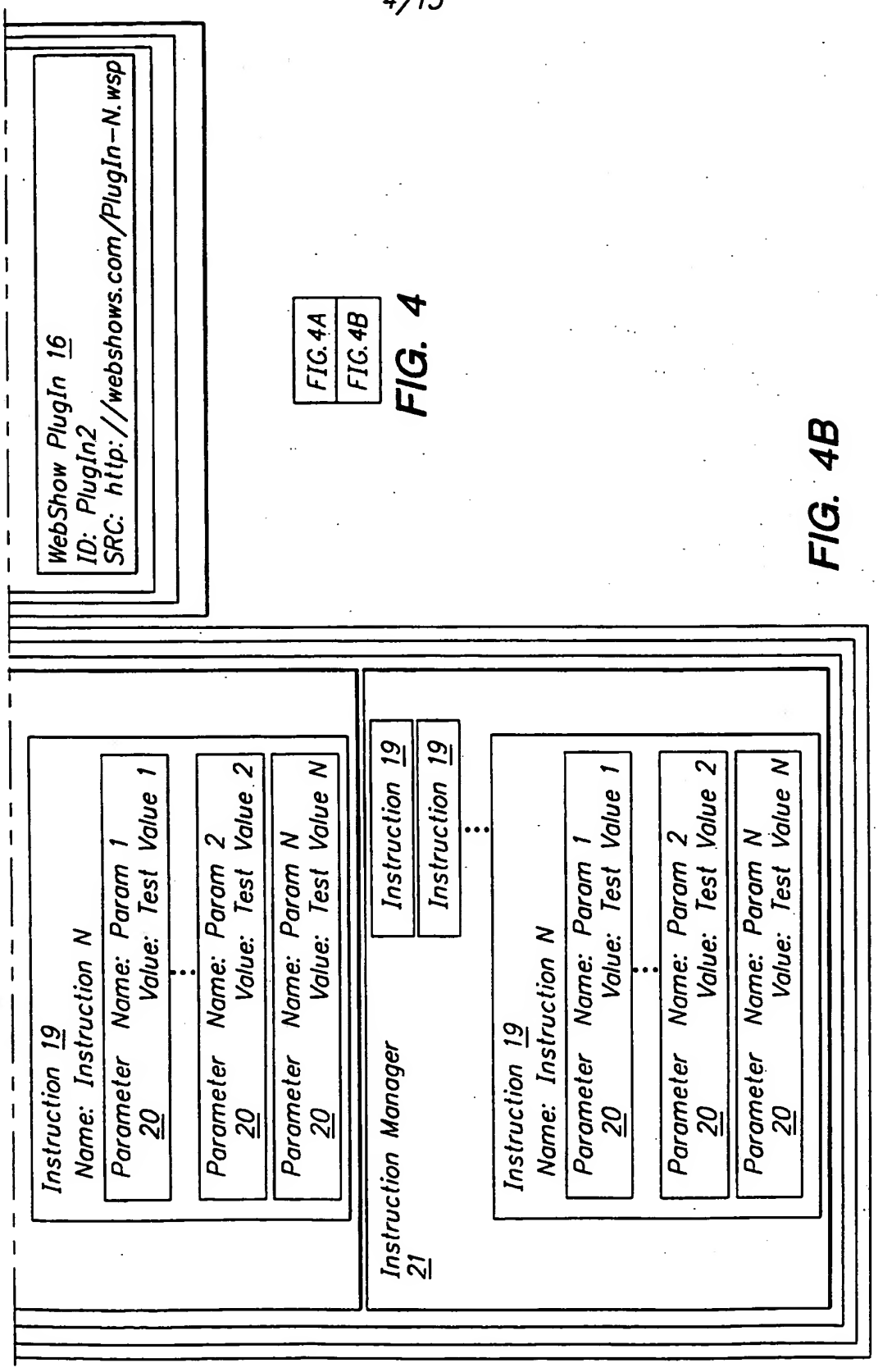


FIG. 4A
FIG. 4B

FIG. 4

FIG. 4B

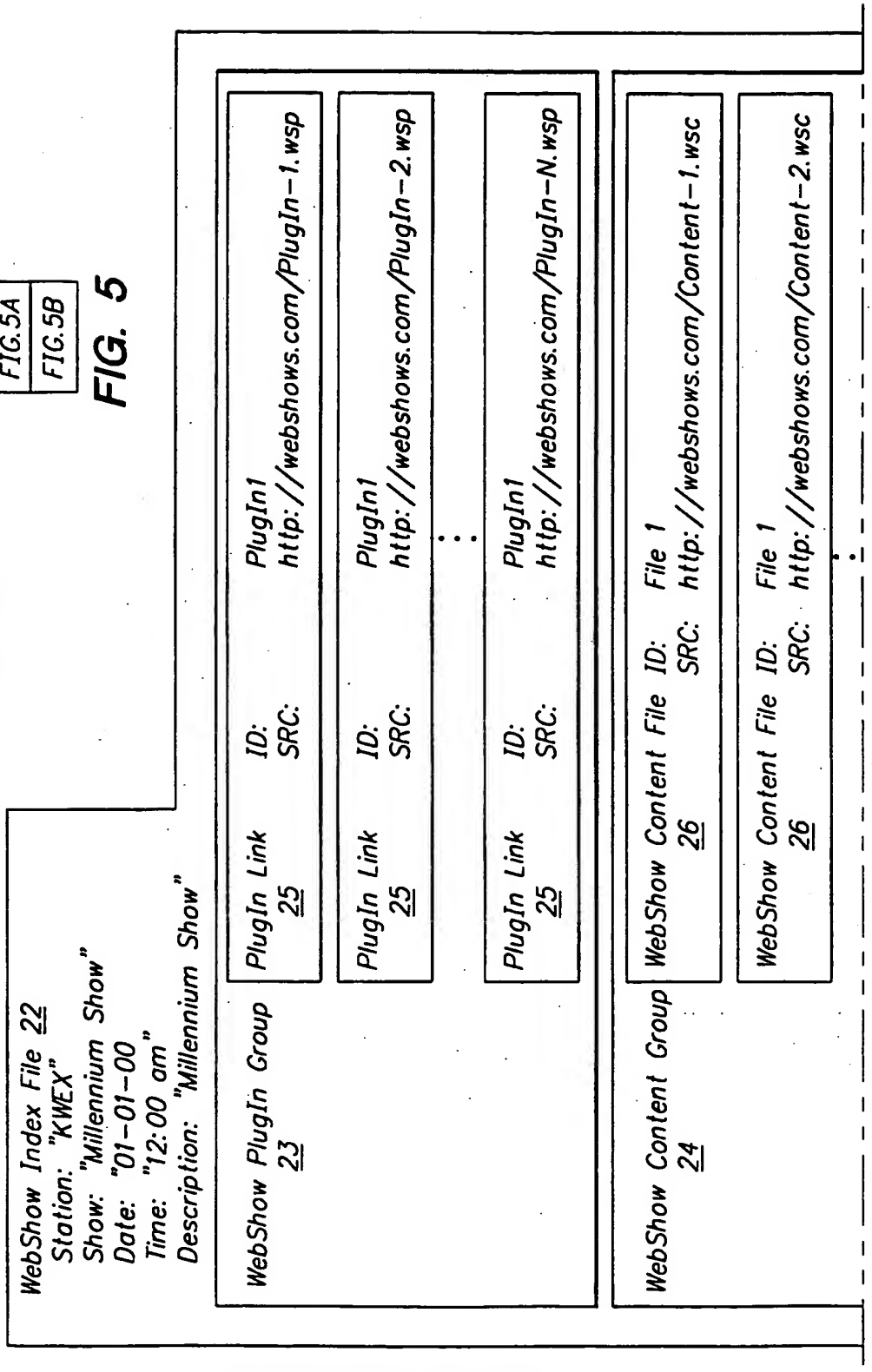


FIG. 5A

FIG.5A
FIG.5B

FIG. 5

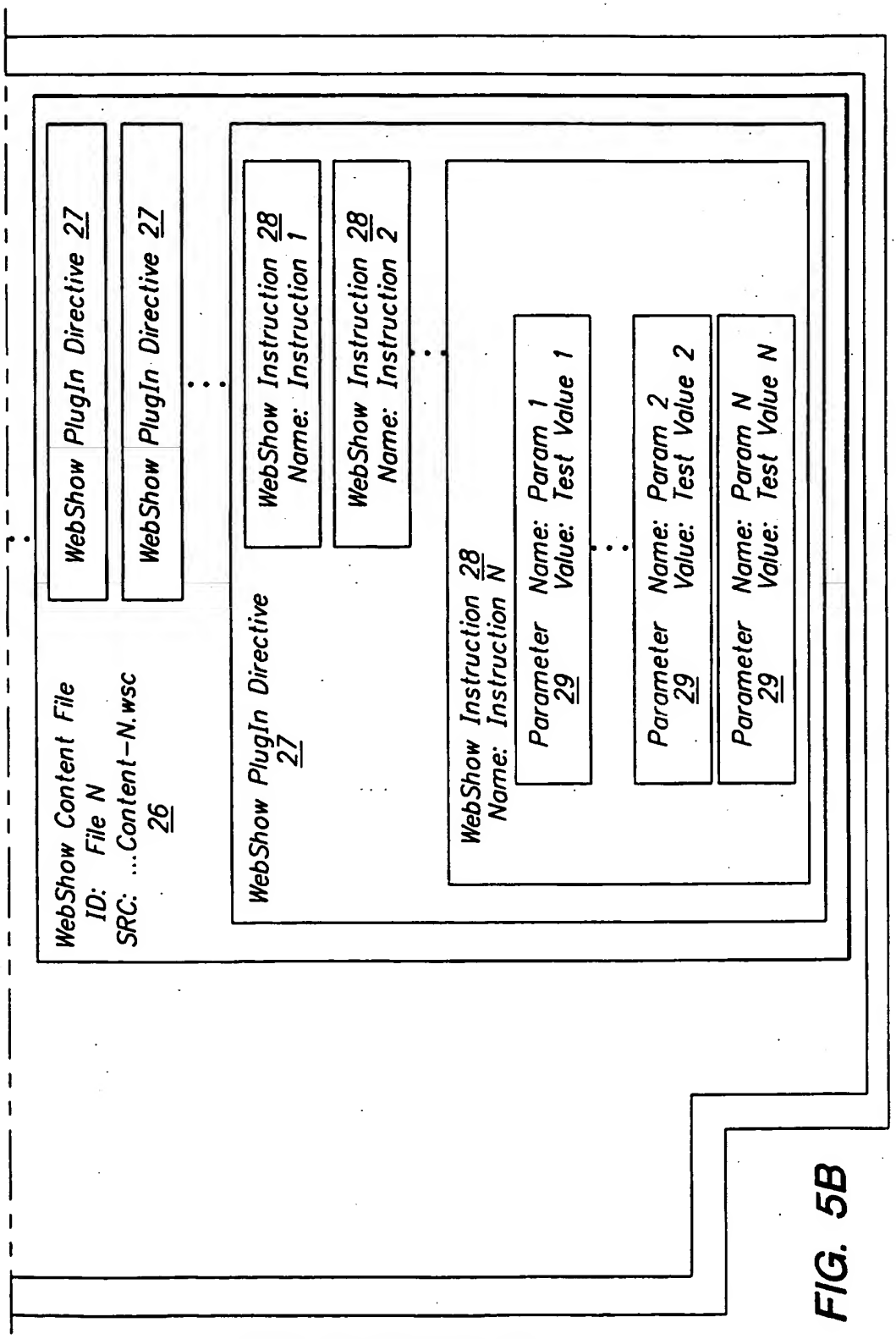

WebShow Content File
ID: File N
SRC: ...Content-N.wsc
26
WebShow PlugIn Directive 27
WebShow PlugIn Directive 27
WebShow PlugIn Directive
27
WebShow Instruction 28
Name: Instruction 1
WebShow Instruction 28
Name: Instruction 2
WebShow Instruction 28
Name: Instruction N
Parameter 29
Name: Param 1
Value: Test Value 1
Parameter 29
Name: Param 2
Value: Test Value 2
Parameter 29
Name: Param N
Value: Test Value N

FIG. 5B

FIG. 6

FIG.6A
FIG.6B

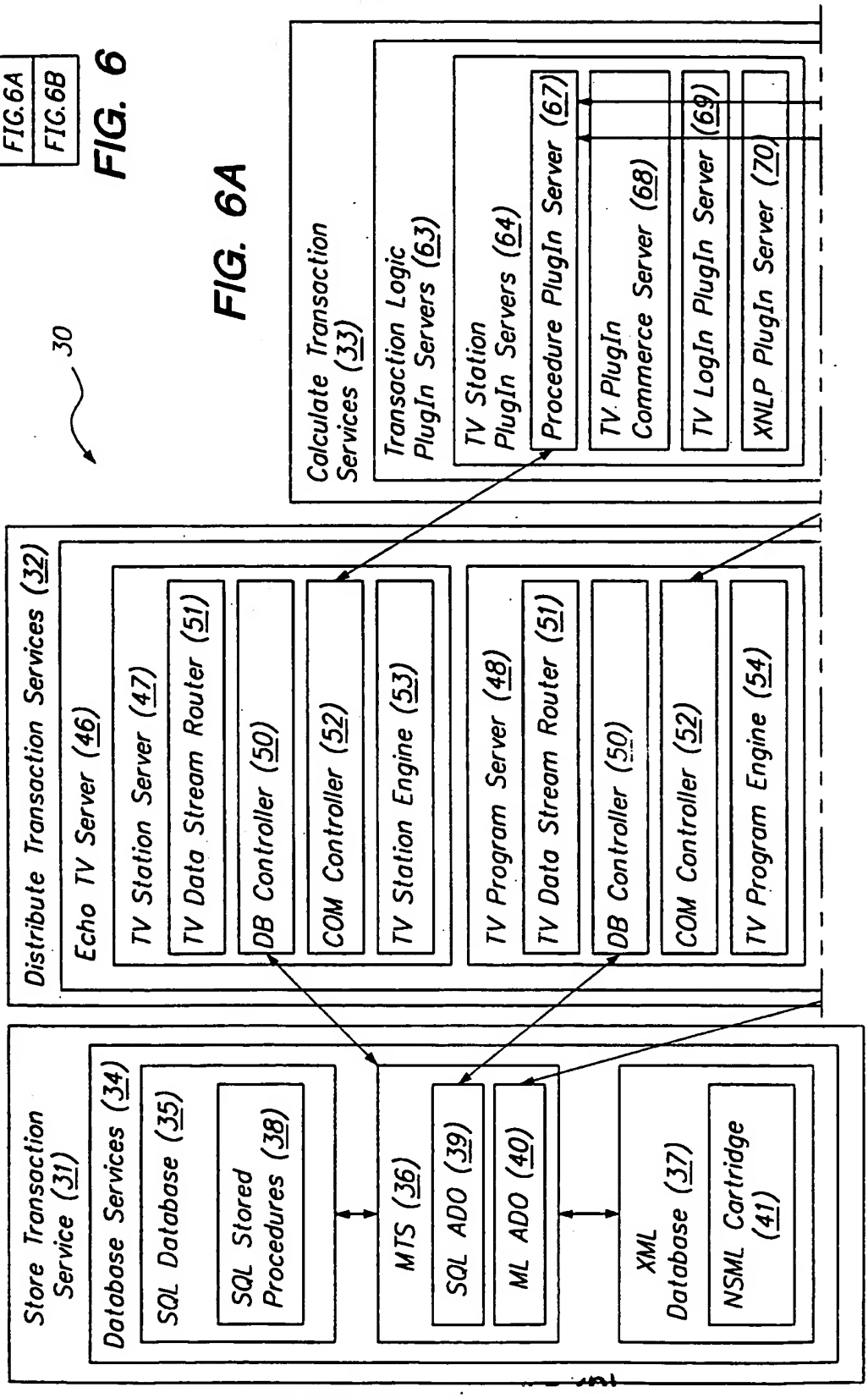


FIG. 6A

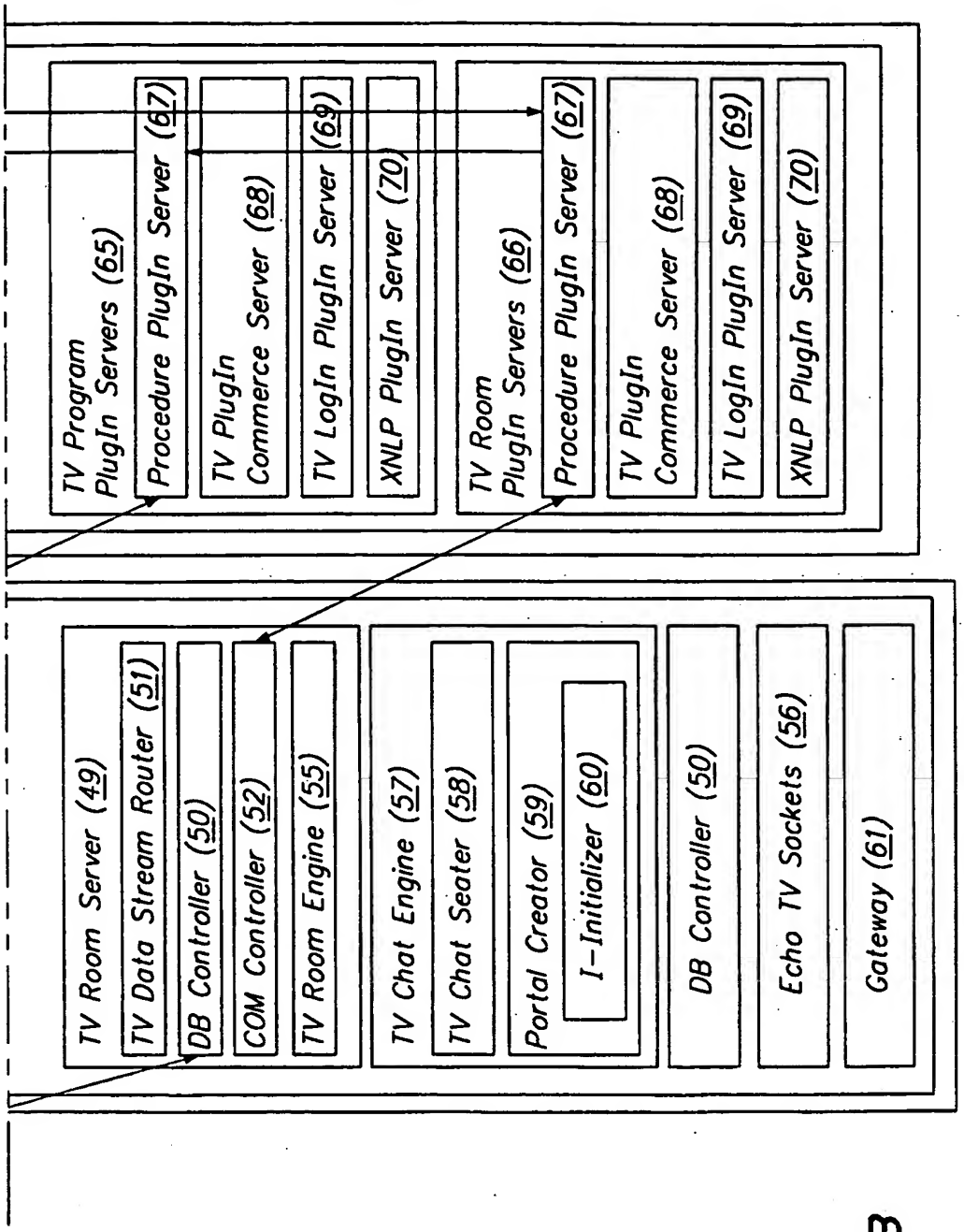

TV Program PlugIn Servers (65)
Procedure PlugIn Server (67)
TV PlugIn Commerce Server (68)
TV LogIn PlugIn Server (69)
XNLP PlugIn Server (70)
TV Room PlugIn Servers (66)
Procedure PlugIn Server (67)
TV PlugIn Commerce Server (68)
TV LogIn PlugIn Server (69)
XNLP PlugIn Server (70)
TV Room Server (49)
TV Data Stream Router (51)
DB Controller (50)
COM Controller (52)
TV Room Engine (55)
TV Chat Engine (57)
TV Chat Seater (58)
Portal Creator (59)
I-Initializer (60)
DB Controller (50)
Echo TV Sockets (56)
Gateway (61)

FIG. 6B

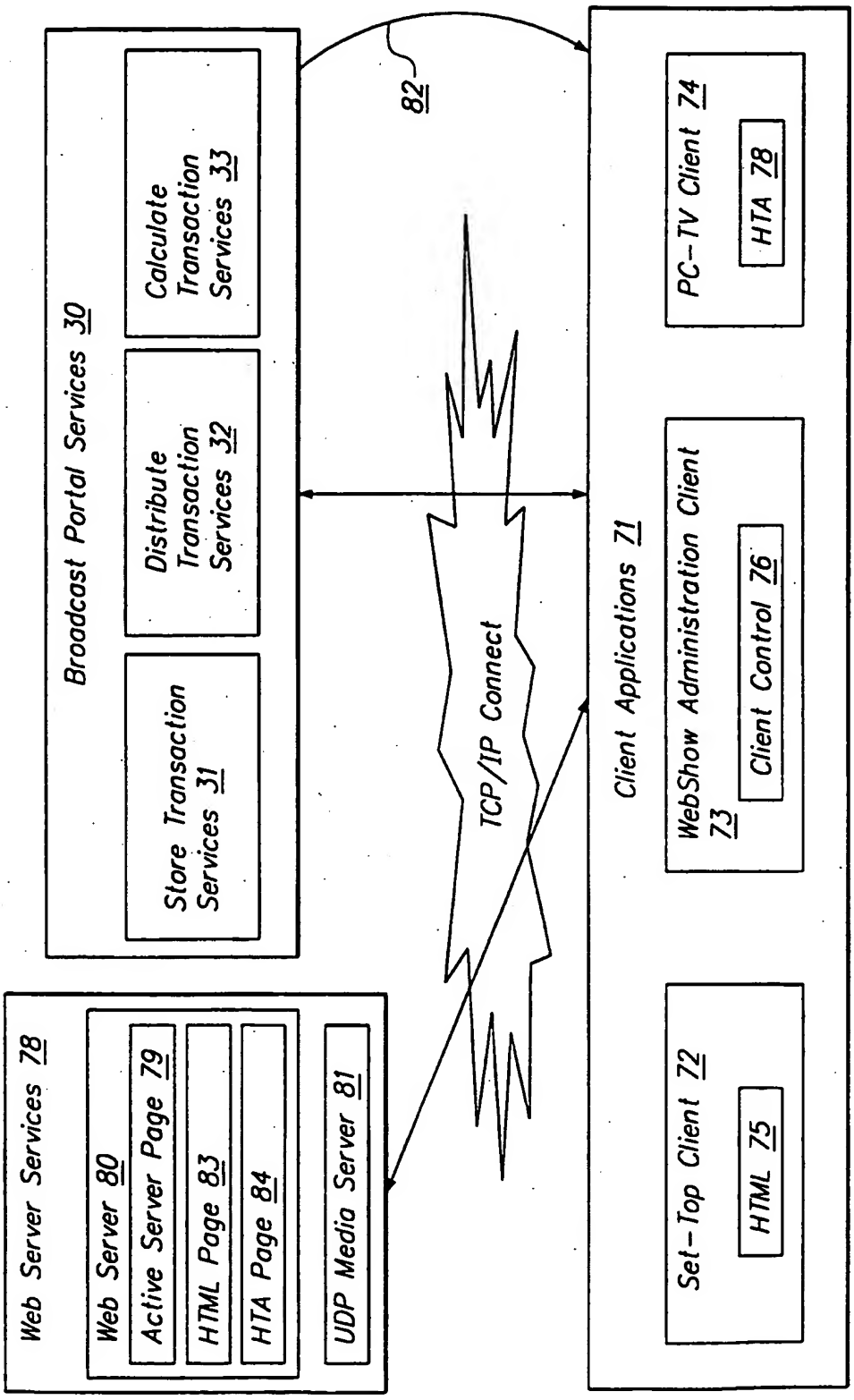

Broadcast Portal Services 30
Store Transaction Services 31
Distribute Transaction Services 32
Calculate Transaction Services 33
82
TCP/IP Connect
Web Server Services 78
Web Server 80
Active Server Page 79
HTML Page 83
HTA Page 84
UDP Media Server 81
Client Applications 71
Set-Top Client 72
HTML 75
WebShow Administration Client 73
Client Control 76
PC-TV Client 74
HTA 78

FIG. 7

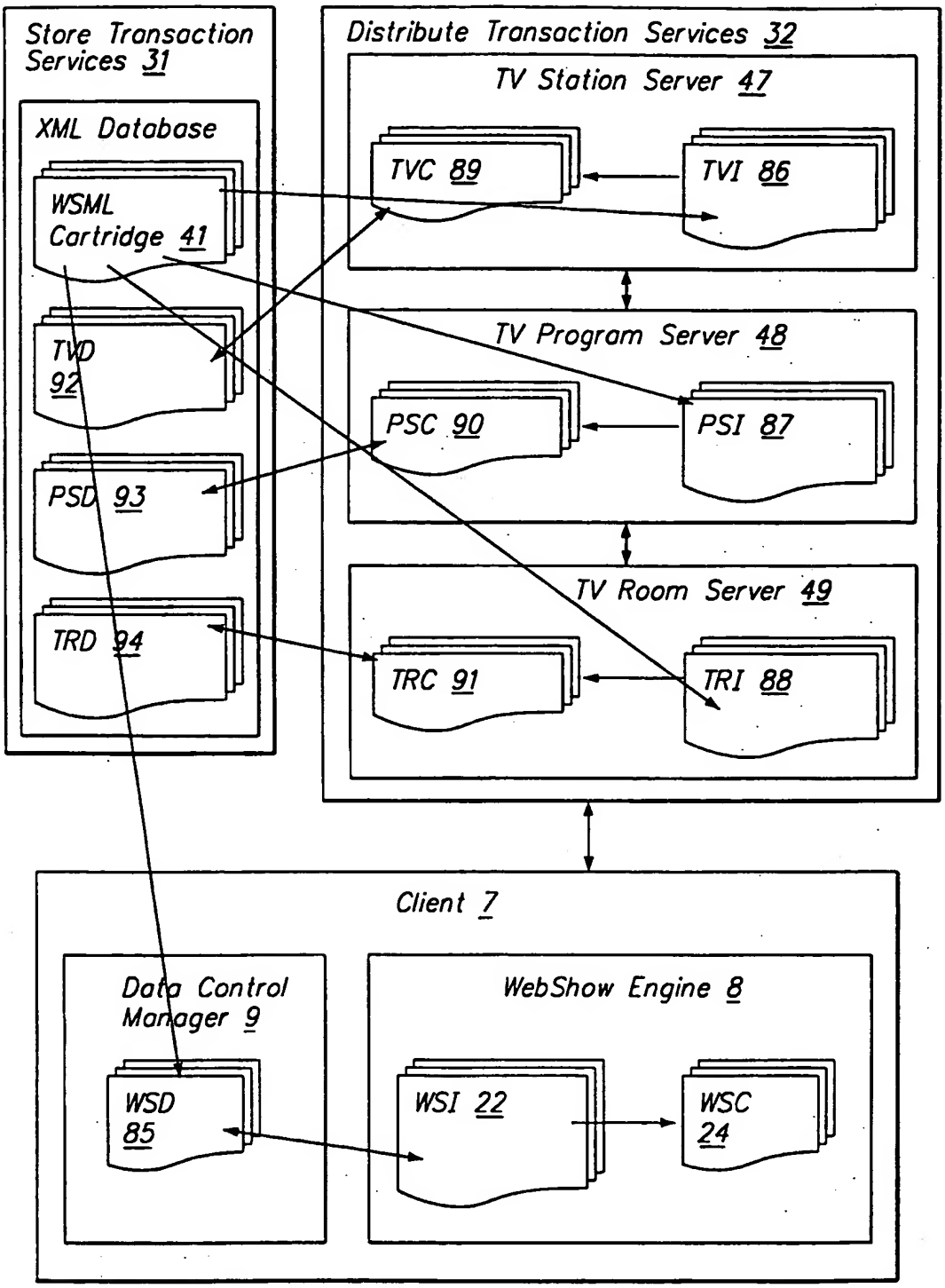

Store Transaction Services 31
XML Database
WSML Cartridge 41
TVD 92
PSD 93
TRD 94
Distribute Transaction Services 32
TV Station Server 47
TVC 89
TVI 86
TV Program Server 48
PSC 90
PSI 87
TV Room Server 49
TRC 91
TRI 88
Client 7
Data Control Manager 9
WSD 85
WebShow Engine 8
WSI 22
WSC 24

FIG. 8

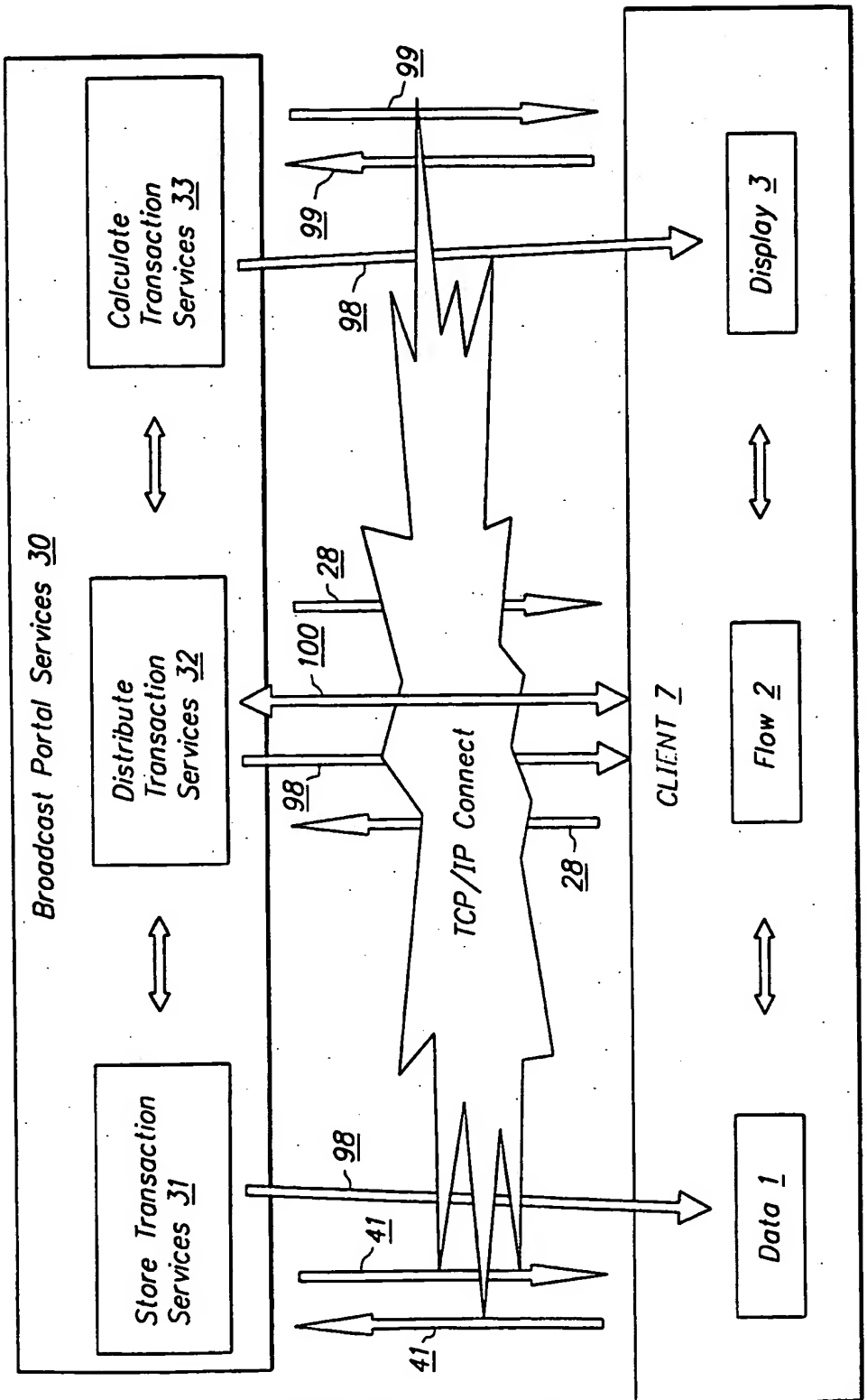

Broadcast Portal Services 30
Calculate Transaction Services 33
Distribute Transaction Services 32
Store Transaction Services 31
TCP/IP Connect
CLIENT 7
Display 3
Flow 2
Data 1
99
98
28
100
41

FIG. 9

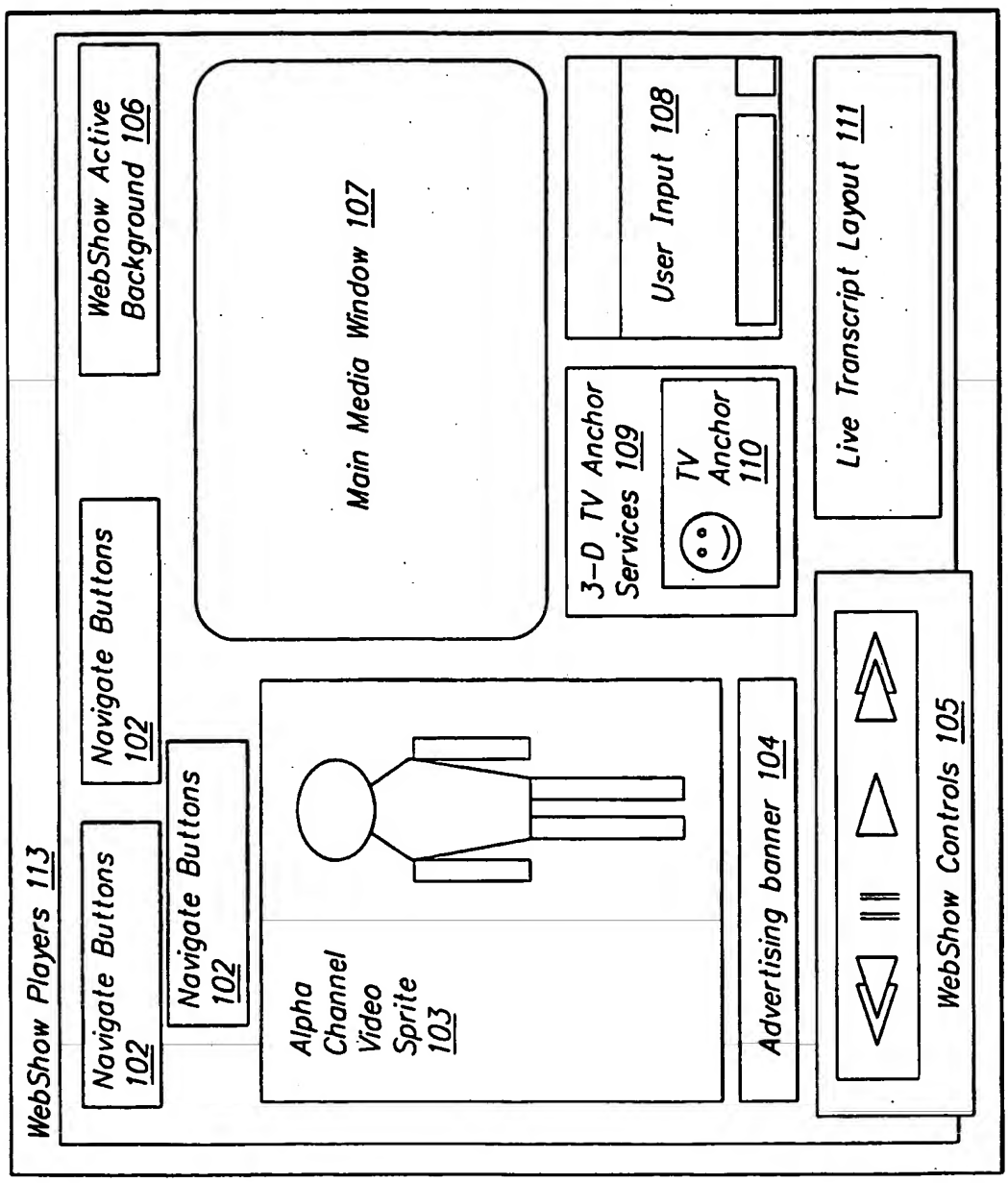

WebShow Players 113
Navigate Buttons 102
Navigate Buttons 102
Navigate Buttons 102
WebShow Active Background 106
Main Media Window 107
Alpha Channel Video Sprite 103
3-D TV Anchor Services 109
TV Anchor 110
User Input 108
Live Transcript Layout 111
Advertising banner 104
WebShow Controls 105

FIG. 10

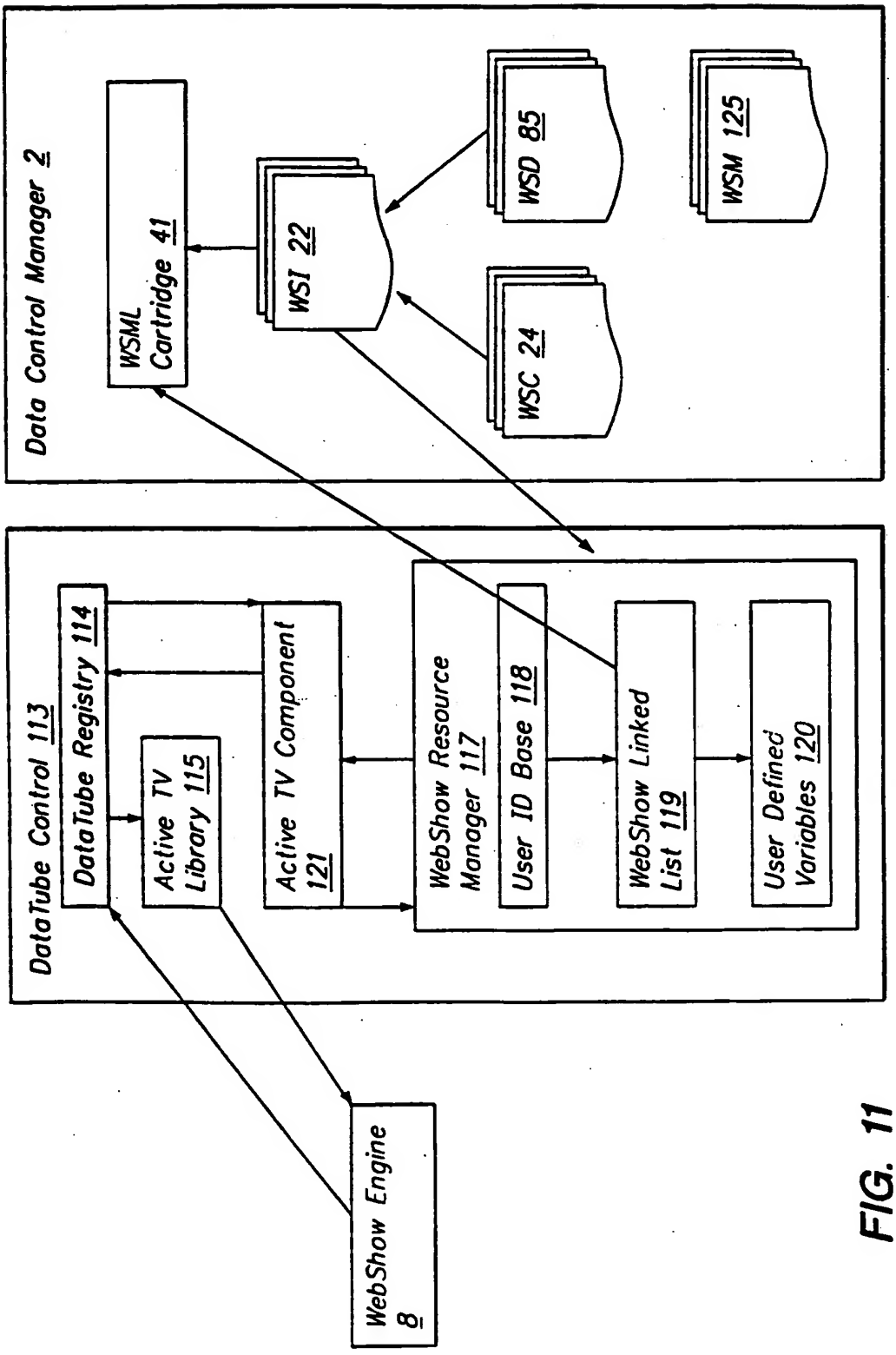

Data Control Manager 2
WSML Cartridge 41
WSI 22
WSD 85
WSC 24
WSM 125
DataTube Control 113
DataTube Registry 114
Active TV Library 115
Active TV Component 121
WebShow Resource Manager 117
User ID Base 118
WebShow Linked List 119
User Defined Variables 120
WebShow Engine 8

FIG. 11

INTERNATIONAL SEARCH REPORT

International application No.
PCT/US00/10499

A. CLASSIFICATION OF SUBJECT MATTER

IPC(7) :H04N 7/12, 7/173, 9/74, ; G11B 31/00; G06F 17/00
US CL :345/327; 348/10, 12, 473, 563, 709/21

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

U.S. : 345/327; 348/10, 12, 473, 563, 709/21

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	US 5,861,881 A (FREEMAN, et al) 19 January 1999; Abstract; col. 2, lines 44-60; col. 7, lines 21-62; col. 9, lines 25-37; col. 19, lines 30-53	1-29
Y,P	US 5,917,830 A (CHEN) 29 June 1999; Abstract; col. 4, lines 45-68; col. 8, lines 21-58	1-29
Y	US 5,754,783 A (MENDELSON, et al) 19 May 1998; Abstract; Fig. 4; col. 4, lines 25-68	1-29
Y	us 5,559,999 A (MATURI, et al) 24 September 1996; Abstract; col. 5, lines 10-55	1-29

☐ Further documents are listed in the continuation of Box C. ☐ See patent family annex.

*	Special categories of cited documents:	"T"	later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention
"A"	document defining the general state of the art which is not considered to be of particular relevance	"X"	document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone
"E"	earlier document published on or after the international filing date	"Y"	document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art
"L"	document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)	"&"	document member of the same patent family
"O"	document referring to an oral disclosure, use, exhibition or other means		
"P"	document published prior to the international filing date but later than the priority date claimed		

Date of the actual completion of the international search 19 JUNE 2000	Date of mailing of the international search report 11 JUL 2000
Name and mailing address of the ISA/US Commissioner of Patents and Trademarks Box PCT Washington, D.C. 20231 Facsimile No. (703) 305-3230	Authorized officer REUBEN BROWN Telephone No. (703) 305-2399

Form PCT/ISA/210 (second sheet) (July 1998) ★